ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ

НОВГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ ЯРОСЛАВА МУДРОГО

Д. В. Михайлов, Г. М. Емельянов

ТЕОРЕТИЧЕСКИЕ ОСНОВЫ ПОСТРОЕНИЯ ОТКРЫТЫХ ВОПРОСНО-ОТВЕТНЫХ СИСТЕМ. СЕМАНТИЧЕСКАЯ ЭКВИВАЛЕНТНОСТЬ ТЕКСТОВ И МОДЕЛИ ИХ РАСПОЗНАВАНИЯ

Монография

ВЕЛИКИЙ НОВГОРОД
2010

УДК 681.3.06
ББК 32.973

Печатается по решению
РИС НовГУ

Р е ц е н з е н т ы

доктор технических наук, профессор **В. В. Геппенер**
(Санкт-Петербургский электротехнический университет)
доктор технических наук, профессор **Ю. Г. Васин**
(Научно-исследовательский институт прикладной математики и кибернетики Нижегородского государственного университета им. Н.И. Лобачевского)

Михайлов Д. В., Емельянов Г. М.

М Теоретические основы построения открытых вопросно-ответных систем. Семантическая эквивалентность текстов и модели их распознавания: Монография / НовГУ им. Ярослава Мудрого. – Великий Новгород, 2010. – 292 с.

В монографии рассматривается задача установления смысловой эквивалентности текстов предметно-ограниченного подмножества естественного языка – важнейшая составляющая организации открытых тестов. Описываются модели, методики и алгоритмы накопления и систематизации знаний о синонимии в естественном языке, выделения семантических отношений на множестве текстов заданной предметной ориентации, привлечения тезауруса предметной области для установления степени близости высказывания смысловому эталону, а также организация такого тезауруса и механизм его пополнения из текстов.

Книга адресована специалистам в области математической лингвистики, анализа данных и знаний, машинного обучения, распознавания и понимания текста.

Монография подготовлена при поддержке УНИК НовГУ.

УДК 681.3.06
ББК 32.973

ВВЕДЕНИЕ

Создание и развитие ЭВМ с расширением сфер их применения привело к потребности приближения языка общения конечного пользователя с ЭВМ к языку решаемой задачи. Появление в 60-е годы специализированных языков программирования высокого уровня для решения задач искусственного интеллекта, с одной стороны, и развитие в 1990-е годы средств автоматизации программирования, с другой стороны, неизбежно ведут к потребности общения пользователя с ЭВМ на предметно-ориентированном языке, максимально приближенном к Естественному Языку (ЕЯ). Разработка таких языков требует моделирования различных аспектов языкового поведения человека в зависимости от задачи, для решения которой разрабатывается тот или иной язык.

Интерес к разработке систем общения с ЭВМ на естественном языке проявляется как со стороны научных дисциплин, так и со стороны технических, связанных с разработкой и программной реализацией интеллектуальных систем. Алгоритмически разрешимые процедуры распознавания смысловых образов высказываний ЕЯ, а также способы представления этих образов, допускающие корректно описываемые процедуры их переработки, позволят программно реализовывать интеллектуальные системы распознавания и синтеза речи, текста и изображений. Разработка таких систем относится к позиции "информационные технологии и электроника" перечня критических технологий федерального уровня от 21 июля 1996 г. и образует самостоятельное направление, получившее название "Обработка естественного языка" [29].

Несмотря на значительный интерес к рассматриваемому направлению, прежде всего различным видам семантического анализа ЕЯ-текстов, на сегодняшний день отсутствует единый теоретический подход к решению практических задач компьютерного моделирования ЕЯ, учитывающий многоуровневость и взаимосвязь всех сторон этого явления. Прежде всего, такое положение обусловлено отсутствием четкого понимания функциональной роли различных сторон языкового поведения при моделировании процессов обработки языковой информации и традиционной ориентацией моделей языка на формальные средства описания и представления знаний. Как следствие этого, большинство из известных на сегодняшний день компьютерных моделей ЕЯ являются редуцированными и недостаточно адекватны моделируемому ЕЯ, будучи разработанными из чисто практических соображений, без привлечения филологических знаний о данном ЕЯ. Сказанное в значительной степени сужает потенциальные возможности построенных на базе этих моделей информационных систем. Ориентация на последние достижения в области экспериментальных исследований языка, прежде всего семантики в ее взаимосвязи с лексикой, синтаксисом и морфологией, моделирование языковых функций, адекватных рассматриваемым ситуациям, дает возможность разрабатывать системы обработки ЕЯ, пригодные для практического применения в решении задач реальной степени сложности.

Сферой рассмотрения авторов предлагаемой Вашему вниманию книги являются задачи, требующие установления полной или частичной эквивалентности по смыслу (Семантической Эквивалентности, СЭ) высказываний (текстов) ЕЯ [108,112]. К числу таких задач можно отнести применение заданий открытой формы в системах компьютерного дистанционного обучения и контроля

знаний [1,71,89], поиск изображений и распознавание семантики сложных информационных объектов по вербальному описанию [80,112,105,106].

С учетом обозначенной выше проблемы компьютерной обработки ЕЯ в плане распознавания смысла, а также практической значимости и ориентации проблемы установления СЭ на применение при построении открытых вопросно-ответных систем, основную цель исследования, которое послужило основой настоящей монографии, можно сформулировать следующим образом:

разработка методов автоматизированного накопления и систематизации знаний о семантической эквивалентности в ЕЯ и алгоритмов, реализующих механизм использования указанных знаний для практических задач анализа текстов.

Для достижения указанной цели в монографии ставятся и решаются следующие задачи:

1). Анализ существующих методов моделирования семантики конструкций ЕЯ и определение общих требований, предъявляемых к механизму сравнения смыслов на функциональном уровне.
2). Разработка и исследование методов моделирования СЭ на уровне варьирования абстрактной лексикой.
3). Разработка методов накопления и систематизации знаний о морфологии и синтаксисе ЕЯ, учитывающих возможные формы выражения заданного смысла.
4). Моделирование и алгоритмизация механизма использования знаний о морфологии и синтаксисе языка для решения задач семантической кластеризации ЕЯ-текстов.
5). Разработка и исследование методов нахождения семантического расстояния между ЕЯ-текстами.

Отсутствие на сегодняшний день единого подхода к описанию ЕЯ, учитывающих всю сложность и многообразие этого явления, недостаточный учет языкового поведения человека в различных видах деятельности разработчиками лингвистических компьютерных систем позволяют констатировать актуальность поставленных и решаемых авторами задач.

Монография включает введение, пять глав, заключение и приложение.

В первой главе осуществляется формулирование требований к процессу сравнения текстов на предмет СЭ. Вводится понятие ситуации языкового употребления и предлагается комплексный подход к решению задачи пополнения лингвистических информационных ресурсов из ЕЯ-текстов с последующим упорядочиванием знаний. При этом особую роль в представлении знаний о синонимии играет уровень глубинного синтаксиса, для которого рассматривается система синонимических преобразований над деревьями глубинных синтаксических структур. Далее в главе рассматриваются достоинства и недостатки установления СЭ на основе указанных синонимических преобразований.

На основе полученного в первой главе теоретико-множественного описания процесса установления СЭ **во второй главе** исследуется проблема полноты представления смысла при описании синонимического варьирования абстрактной лексикой на уровне глубинного синтаксиса. Ставится и решается задача построения формальных семантических образов сверхфразовых единств на указанном языковом уровне в ЕЯ-высказываниях, состоящих более чем из одного простого распространенного предложения. Рассмотрены вопросы эффективного использования знаний уровня глубинного синтаксиса при решении указанной задачи.

Третья глава посвящена вопросам автоматизированного получения знаний, необходимых для установления синонимии в рамках подхода "Смысл⇔Текст", рассмотренного в первых двух главах. С целью формализации условий применимости правил синонимических преобразований деревьев глубинного синтаксиса предложено описание толкования лексического значения слова на языке логики предикатов 1-го порядка. Исследованы принципы обобщения независимых вариантов формализованных толкований значения слова относительно заданного предметно-ориентированного подмножества ЕЯ. Для автоматизации получения толкований значений слов как основы формирования условий применимости синонимических преобразований предложена комплексная методика выделения и классификации отношений, необходимых для построения тезауруса предметной области и ролевой идентификации сущностей относительно заданной ситуации. В качестве исходных данных для выделения указанных отношений предложено использовать множества СЭ-фраз, в составе каждого из которых ЕЯ-фразы описывают одну и ту же ситуацию действительности.

В четвертой главе монографии рассматривается использование синтаксического контекста существительного в качестве основы семантической кластеризации текстов. На основе свойств соотношения смыслов в составе синтаксического контекста соподчиненных слов решается задача установления частичных СЭ.

В пятой главе синтаксический контекст существительного анализируется в рамках ситуации языкового употребления. Для последней строится математическая модель в виде формального контекста. Описывается методика его построения на основе множества семантически эквивалентных фраз предметно-ориентированного подмножества естественного языка. Далее в главе

вводится мера схожести между формальными контекстами ситуаций языкового употребления и описываются правила установления семантической эквивалентности фраз относительно заданного предметно-ориентированного подмножества естественного языка.

В приложении приведены фрагменты исходного текста программы, осуществляющей генерацию модели ситуации языкового употребления на основе множества семантически эквивалентных ЕЯ-фраз. Описывается реализация на языке Visual Prolog 5.2 процедур выделения основ посимвольным сравнением слов различных фраз, таксономии буквенных инвариантов в составе отдельных слов при выявлении основ с учетом возможных синонимов и частичных совпадений буквенного состава основ у слов с разным лексическим значением, а также методов оценки качества такой таксономии.

Описанные в монографии методы, модели и алгоритмы могут представлять практический и научный интерес для специалистов в области текстового анализа, а также смежных с ним направлений распознавания и анализа семантики сложных информационных объектов, в частности, морских навигационных карт. Книга может быть полезной для студентов и аспиратов математических, филологических и инженерных специальностей. Полученные лично авторами и представленные в монографии научные результаты и программное обеспечение широко используются в совместной с ВЦ РАН научно-учебной лаборатории распознавания образов и обработки изображений при подготовке выпускных квалификационных работ, диссертаций, чтении спецкурсов в Новгородском государственном университете имени Ярослава Мудрого.

Глава 1

АВТОМАТИЧЕСКАЯ КОМПРЕССИЯ ТЕКСТОВ И РАСПОЗНАВАНИЕ СМЫСЛОВОЙ ЭКВИВАЛЕНТНОСТИ

Настоящая глава посвящена общей постановке задачи сравнения текстов на предмет эквивалентности смыслов. Формулируются общие требования к процессу установления семантической эквивалентности текстов относительно предметно-ограниченного подмножества естественного языка. Показывается, что использование знаний о синонимии позволяет уменьшить объем памяти ЭВМ, необходимой для хранения текстовой информации. Строится теоретико-множественное описание процесса установления семантической эквивалентности с учетом выявленных функциональных требований. На основе теории анализа формальных понятий предложен комплексный подход к решению задачи пополнения лингвистических информационных ресурсов из текстов с последующим упорядочиванием знаний.

1.1. Семантическая эквивалентность и ситуация языкового употребления

В настоящий момент в теоретической лингвистике и смежных с ней дисциплинах не существует общепризнанного и бесспорного определения языка как такового.

В частности, существует довольно распространенное понимание языка как сложной знаковой системы [3,31,88]. Различные знаковые

системы являются предметом изучения семиотики [32,40]. При этом сам естественный язык рассматривается с двух точек зрения [83].

С функциональной точки зрения строение ЕЯ определяется использованием последнего в качестве средства общения. Формальная точка зрения предполагает наличие у языка некоторой абстрактной модели, которая не зависит от конкретного способа использования ЕЯ и может быть описана формальной грамматикой. Моделирование естественных языков с помощью формальных грамматик, порождающих возможные высказывания, было предложено Н. Хомским [87]. Хорошим примером рассмотрения языка с функциональной точки зрения может послужить модель языка как преобразователя "Смысл⇔Текст" [45].

Совмещая точки зрения и подходы к моделированию, естественный язык следует определить как сложную знаковую систему, основной функцией которой является использование в качестве средства общения между людьми. При этом абстрактная модель языка задается формальным механизмом порождения всех возможных высказываний в этой знаковой системе, а также механизмом установления соответствия высказываниям их смыслов плюс установление соответствия между самими смыслами. Под естественностью языка будем понимать наличие таких свойств как синонимия слов и словосочетаний, а также свободный порядок слов в предложении [66,67,74].

Опираясь на данное определение ЕЯ, введем некоторые базовые термины для формального описания рассматриваемой нами задачи установления СЭ.

Определение 1.1. Под *конструкцией ЕЯ* (далее в работе мы будем также использовать термин *языковая конструкция*) в настоящей работе понимается последовательность знаков в некоторой

Знаковой Системе, которая может быть использована для фиксации некоторого количества высказываний этого ЕЯ в памяти ЭВМ.

Определение 1.2. Семантическими знаниями мы будем называть языковые знания, необходимые для использования некоторого ЕЯ в процессе общения. Соответственно, *носителем языка* следует считать обладателя семантических знаний.

Следствие. Под *семантическим отношением* следует понимать некоторую универсальную связь, усматриваемую носителем языка в тексте. Именно таким образом понимается семантическое отношение в идеологии Русского Общесемантического Словаря (РОСС) [41].

Смысл высказывания представляет собой довольно сложный и удаленный от уровня наблюдения конструкт [69].

Строгое формальное определение смысла, которое авторами настоящей монографии использовано в рамках предлагаемых моделей, методик и алгоритмов, будет дано в главе 3. Здесь мы остановимся на следующем определении в первом приближении, приемлемом с точки зрения практики обработки текста.

Определение 1.3. Под *смыслом* высказывания понимается информация, содержащаяся в высказывании и подлежащая передаче и восприятию. Иными словами, смысл – это информация о том, как объект или ситуация реального мира отражается в сознании говорящего.

Рассматривая текст как поверхностную форму фиксации высказываний на ЕЯ и единственный способ выражения смысла в процессе общения с ЭВМ на этом ЕЯ, то есть допуская знаковую систему в качестве единственного средства выражения смысла, будем считать, что понятие смысловой эквивалентности совпадает с понятием семантической эквивалентности.

При этом задача установления семантической эквивалентности ЕЯ-высказываний состоит в сравнении информации, отвечающей *Определению 1.3*, посредством обработки конструкций ЕЯ, которые эту информацию фиксируют [76]. Семантическую эквивалентность, таким образом, в общем случае следует понимать как теоретико-множественное пересечение смыслов.

Исходя из сформулированного нами определения ЕЯ как сложной знаковой системы, в качестве модели семантики конструкций ЕЯ для решения задачи установления СЭ будем использовать модель ситуаций употребления ЕЯ.

Предназначение таких ситуаций состоит в разделении языкового опыта в соответствии с разделением концептуальной картины мира. Подобное разделение лежит в основе генезиса ЕЯ [44]. Ситуации языкового употребления рождаются из потребности обозначить и описать новый социальный опыт либо содержание обстоятельств типичных совместных действий [73] посредством ЕЯ.

Определение 1.4. Под ситуацией языкового употребления (ситуацией употребления ЕЯ) понимается описание нового социального опыта (содержания совместных действий) средствами этого ЕЯ. Данное описание выполняется в некоторой знаковой системе с целью обобщения и передачи знаний от человека к человеку.

Формально фиксируемый ситуацией S языковой контекст представляется тройкой:

$$S = (O, R, T), \tag{1.1}$$

где O есть множество объектов-участников S, R – множество отношений между $o \in O$, T – множество форм языкового описания S.

Следует отметить, что посредством модели (1.1) могут быть представлены любые семантические знания о заданном ЕЯ.

Действительно, конкретный вид элементов множества T не определен, что позволяет представлять формы языкового описания S, в частности, деревьями синтаксического подчинения. А поскольку синтаксические отношения задают синтагматические зависимости, которые определяют возможность сосуществования словоформ в линейном ряду, то допускается приводить элементы множества T к естественному для поверхностного уровня ЕЯ представлению в линейной форме.

В то же время отношения из множества R также могут быть любого типа, что позволяет описывать посредством модели (1.1) любые преобразования конструкций заданного ЕЯ.

Согласно *Следствию Определения 1.2*, синтаксические зависимости можно рассматривать как частный случай семантических отношений, что дает возможность решать задачу формирования и классификации синтаксических отношений относительно различных ситуаций вида (1.1). Этот вопрос более подробно освещается в пятой главе работы.

Модель (1.1) учитывает как парадигматические отношения, которые задаются с помощью варьирования объектов множества O, так и синтагматические отношения между языковыми единицами. Последний вид отношений задается с помощью множества R.

Кроме того, смысл ситуации S отделен от множества форм поверхностного выражения данной ситуации. Благодаря такому разделению допускается сравнение смыслов без порождения всех возможных инвариантных по смыслу фраз.

1.2. Концептуальная модель процесса установления семантической эквивалентности

На основе введенного в предыдущем разделе представления о ситуации языкового употребления как основы моделирования семантики ЕЯ в задаче установления СЭ, настоящий раздел имеет целью описание данной задачи не функциональном уровне и установление границ проблемной области сравнения смыслов.

В общем случае задача СЭ формулируется следующим образом.

Задача 1.1. Дано множество ЕЯ-текстов G. Элементами этого множества могут быть, к примеру, развернутые ответы обучаемых на вопрос тестирующей системы при применении заданий открытой формы. Требуется: по результатам разбора каждого $T_i \in G$ выявить:

- множество $V(T_i)$ ситуаций, описываемых T_i;
- множество $M(T_i)$ объектов и/или понятий, значимых в ситуациях из множества $V(T_i)$;
- тернарное отношение $I \subseteq G \times M \times V$, ставящее в соответствие каждому $m \in M : M = \bigcup_i M(T_i)$ ту ситуацию $v \in V : V = \bigcup_i V(T_i)$, в которой он фигурирует относительно T_i.

Далее на основе выявленного отношения I необходимо выделить группы текстов, сходных по встречаемости объектов в одних и тех же ситуациях. В конечном итоге требуется доказать идентичность ролей сходных понятий относительно сходных ситуаций, описываемых сравниваемыми текстами.

Наиболее близка данной идее обработка текстов на основе коммуникативной грамматики. Хорошим примером является поисковая система Exactus [84].

Тем не менее, существуют задачи сравнения смысла, отличные от традиционного для поисковых систем взаимодействия "запрос-ответ".

Примером является интерпретация текста ответа на тестовое задание открытой формы в системе автоматизированного контроля знаний. Необходимо не столько отобразить ответ на предметную область, сколько оценить его близость ответу, "правильному" с точки зрения преподавателя, конструировавшего тест. Анализ близости высказываний здесь требует учета лексико-функциональной синонимии, в частности – расщепленных значений и конверсивов [45]. В более общем случае многих обучаемых мы имеем задачу текстовой кластеризации [123].

По оценке Г.С. Осипова [68], требуется более детальное исследование свойств семантических связей и в самой коммуникативной грамматике.

Как следует из определения, сформулированного нами в предыдущем разделе главы, задача установления СЭ условно разбивается на две подзадачи: задачу восприятия текстов и задачу сравнения семантических представлений входных текстов.

Согласно *Определению 1.2*, процесс решения задачи восприятия текста строится на семантических знаниях. Задача сравнения семантических представлений решается посредством той части абстрактной модели языка, которая обеспечивает соответствие между смыслами (переход от одного семантического представления к другому).

К основным проблемам при выбранном нами теоретико-философском подходе можно отнести следующее.

Во-первых, модель вида (1.1) должна быть по-настоящему формализована. Формализация модели здесь означает, в частности,

формализацию представления каждой из ее компонент для различных языковых уровней, поскольку природа отношений из множества R не ограничена. Кроме того, необходима формализация взаимосвязей и между самими языковыми уровнями, что в модели (1.1) не учитывается.

Во-вторых, необходимо разработать способы представления самих семантических знаний в системе и механизмы их пополнения. Семантические знания являются той базой, которая обеспечивает решение как задачи восприятия текста, так и задачи сравнения семантических представлений.

Основные достоинства выбранного подхода можно сформулировать следующим образом. Организация систем обработки ЕЯ-текстов на базе семантических знаний позволяет расширить возможности этих систем от жесткой ориентации на работу в предельно ограниченной предметной области. Это объясняется тем, что центральное место в семантических исследованиях большинства лингвистических теорий занимает не конкретная предметная лексика, а абстрактные слова (названия отношений, слова-кванторы), за счет которых обеспечивается богатое варьирование форм языкового описания для ситуации вида (1.1). Именно абстрактные слова должны в первую очередь подвергаться семантическому анализу [3,45,30].

Рассмотрим теперь, какие из выделенных нами требований к функционированию системы установления СЭ являются ключевыми для оценки адекватности рассматриваемых далее в работе моделей.

В общих чертах следует считать, что относительно заданного предметно-ориентированного подмножества ЕЯ модель решает задачу установления СЭ, если она устанавливает семантическое тождество внешне различных предложений (синонимию) и анализирует грамматическую правильность предложений. В более общем случае

отсутствия предметных ограничений модель должна также устанавливать семантическое различие внешне совпадающих предложений (омонимию).

Для решения задачи формального описания отношений синонимии и омонимии между предложениями ЕЯ и задачи распознавания грамматической правильности предложений необходим формальный аппарат лингвистических описаний [112]. Если естественный язык представить в виде формальной системы, то, согласно принятой нами идее семантики конструкции ЕЯ, он становится языком описания смыслов в формальной модели семантической эквивалентности. Подробнее об описании смыслов языковых конструкций на самом естественном языке мы остановимся в *Главе 3*. Сейчас же мы сформулируем основные требования к языку формального описания и исчисления смыслов для задачи СЭ.

Во-первых, каждый комбинаторный тип цепочки в таком языке должен иметь один и только один смысл. При наложении ограничений предметного характера фразы ЕЯ при единственности синтаксической интерпретации могут обладать множественностью семантических интерпретаций, соответствующих смысловым оттенкам, но не нести взаимно исключающие смыслы. В этом случае понимание обеспечивается пресуппозицией [66].

Во-вторых, язык описания и исчисления смыслов должен быть языком универсальной канонизации, то есть накладываемые на язык ограничения не зависят от предметной области, которую этот язык описывает.

При этом сама модель СЭ должна быть такова, что любой ее компонент не только может быть реализован на ЭВМ, но и способен к расширению в автоматическом режиме, то есть на основе входных текстов. Иными словами, модель должна быть динамической.

1.3. Уровень глубинного синтаксиса

В наибольшей степени требованиям, отмеченным в предыдущем разделе, отвечает модель языка как преобразователя "Смысл⇔Текст" [45]. Действительно, сам естественный язык в данном теоретическом подходе рассматривается как преобразователь текстов в смыслы и обратно. При этом смысл рассматривается как инвариант синонимических преобразований одних конструкций ЕЯ в другие, что позволяет выстраивать иерархию синонимических преобразований, решая задачу установления соответствия между смыслами. Предполагается, что сама синонимия языковых конструкций возникает не только за счет лексических синонимов, но и за счет синтаксических и лексически обусловленных вариантов высказывания.

В модели "Смысл⇔Текст" эти средства представлены в виде синтаксических и лексических правил перифразирования, базирующихся на аппарате Лексических Функций (ЛФ) [45].

Как отмечал И.А. Мельчук, "каждая ЛФ есть функция в математическом смысле, представляющая некоторый весьма общий смысл типа 'очень', 'начинаться' или 'выполнять', или же определенную семантико-синтаксическую роль ("быть подлежащим, будучи первым актантом в данной ситуации" и т.п.)" [118].

Иными словами, лексическая функция показывает смысловую связь слова с другими словами, способными либо замещать его в тексте при определенных условиях, либо образовывать с ними фразеологические сочетания. При этом богатое словесное варьирование присуще только небольшому числу смыслов, которые и выделяются в качестве стандартных лексических функций-

параметров. Данный вид синонимии, именуемый в литературе как ЛФ-синонимия, имеет следующие особенности:

– Глубинным Синтаксическим Структурам (ГСС) сравниваемых высказываний соответствуют одни и те же (или эквивалентные) Семантические Представления (СемП, [45, стр. 32]);

– в Семантическом Графе (СГ) СемП выделяются подграфы (пучки) и каждому подграфу СГ будет соответствовать свое поддерево ГСС каждого из сравниваемых высказываний;

– существует как минимум один подграф СГ, который будет по-разному отображаться в глубинных синтаксических структурах каждого из сравниваемых высказываний. Иными словами, один и тот же смысл в разных ГСС выражается разными обобщенными лексическими единицами [45, стр. 178] рассматриваемого ЕЯ. Но при этом перераспределение смысла между лексемами, как показано в [45, стр. 147], сводится к минимуму, а смысловые соотношения между цельными лексическими единицами описываются с помощью аппарата стандартных ЛФ.

Как отмечено в [45, стр. 147], в силу регулярности стандартных ЛФ и операций над ними, ЛФ-синонимические отношения между ГСС оказываются более регулярными и однотипными, нежели чем произвольные синонимические отношения между ГСС. ЛФ-синонимические отношения между ГСС могут быть описаны с помощью специального исчисления в виде системы правил, которая любой данной ГСС ставила бы в соответствие все другие ГСС, ЛФ-синонимичные с ней. При этом саму задачу установления СЭ можно переформулировать следующим образом.

Задача 1.2. Дано:

Π^R – множество правил ЛФ-синонимических преобразований;

L^{Π} – множество пар ЕЯ-высказываний, между которыми возможно установление синонимии (относительно Π^R);

$r(\pi)$ – условие применимости правила $\pi \in \Pi^R$. Для $L_i = \{T_1, T_2\}: L_i \in L^{\Pi}$ $r(\pi)$ есть совокупность требований к $\forall w_j \in W, W = W_1 \cup W_2$, где $W_1 \subset T_1$, $W_2 \subset T_2$, а W_1 и W_2 – совокупности слов, заменяемых посредством π.

Требуется: для произвольной пары L_k ЕЯ-высказываний проанализировать условие применимости каждого правила множества Π^R и выделить образ класса $\pi \in \Pi^R$, на который объект L_k наиболее похож. При этом $r(\pi)$ выступает в качестве прецедента как типичного представителя таксона π.

Данная задача является классической задачей Распознавания Образов [28]. Использование прецедентов при таком подходе позволяет сократить объем памяти, необходимой для хранения текстовых баз данных при рассмотрении текстов как сложных информационных объектов с внутренней структурой [112]. Сказанное, в частности, актуально для поисковых систем [84].

Действительно, для каждого текста необходимо выделить его класс СЭ, который соответствует $r(\pi)$. Далее происходит поиск уже внутри данного класса того подкласса, который наиболее соответствует данному тексту и включает тексты, максимально синонимичные заданному. По сути, данные о текстах будут описываться некоторой иерархической структурой, каждый новый текст будет определяться только теми признаками, которые отличают этот текст от других представителей наиболее близкого ему класса. Причем в процессе поступления новых текстов в базу классификационные признаки будут постоянно уточняться уже за

рамками лексико-функциональной синонимии. Выделение подклассов СЭ при этом производится согласно постановке *Задачи 1.1*, сформулированной нами в начале раздела 1.2. Подклассы СЭ будут соответствовать смысловым оттенкам как отдельных слов, так и высказываний в составе языковых конструкций. Заметим, что элементы множества форм языкового описания одной и той же ситуации (1.1) в общем случае вполне могут относиться к различным классам СЭ (относительно различных $r(\pi)$).

Формирование знаний, соответствующих лексико-функциональной синонимии относительно Π^R и текстовой кластеризации относительно описываемых текстами объектов (понятий) и ситуаций будет рассмотрено нами в третьей и четвертой главах, соответственно. Сейчас же мы остановимся более подробно на механизме установления соответствия высказываниям их смыслов для модели “Смысл⇔Текст” как абстрактной модели языка.

Поскольку в модели “Смысл⇔Текст” смысл рассматривается как инвариант всех синонимических преобразований, то семантику следует рассматривать как совокупность правил преобразований одних конструкций ЕЯ в другие конструкции, эквивалентные им по смыслу. Сам смысл при этом задается с помощью формального языка, включающего помимо грамматического компонента и правил перевода конструкций ЕЯ в выражения на языке смыслов, процедуру разрешения проблемы эквивалентности языковых конструкций как на уровне ЕЯ, так и на уровне формализованного описания смыслов [108].

Если ограничить рассмотрение синонимии только ЛФ-синонимией, то, как показано в [112], в роли указанного формального языка будет выступать язык глубинного синтаксиса, а в качестве

формального аппарата моделирования СЭ – грамматики деревьев (Δ-грамматики [7,8,112]) вида:

$$\Gamma^{RL} = \left(W^R, V^R, \varphi, \Pi^R, \Phi^R\right), \qquad (1.2)$$

именуемые в [112] расширенными универсальными правильными Δ-грамматиками.

Здесь W^R есть конечное множество (словарь) пометок на узлах, V^R – конечное множество (словарь) пометок на ветвях деревьев. φ есть суть отображение множества V^R во множество натуральных чисел, представляемое в матричной форме как

$$\varphi = \begin{pmatrix} a_1 & a_2 & \dots & a_k \\ n_1 & n_2 & \dots & n_k \end{pmatrix}, \text{ где} \qquad (1.3)$$

$V^R = \{a_1, a_2, \dots, a_k\}$, а $\{n_1, n_2, \dots, n_k\}$ – подмножество натуральных чисел. Причем предполагается, что все деревья удовлетворяют следующему ограничению: для $\forall i = 1, \dots, k$ из любого узла дерева выходит не более $\varphi(a_i) = n_i$ ветвей с пометкой a_i. В этом случае также говорят, что дерево является φ-правильным [112].

Применительно к ЛФ-синонимическим преобразованиям глубинных синтаксических структур компонент Π^R в (1.2) получает содержательную интерпретацию множества синтаксических, а Φ^R - множества вспомогательных лексических правил преобразований деревьев глубинного синтаксиса. Множество V^R становится множеством типов глубинно-синтаксических отношений, $V^R = \{1,2,3,4,5,6\}$ [45]. Матрица (1.3) здесь отражает характер ограничений на ветвление в реальных ГСС: $\varphi = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 1 & 1 & 1 & 4 & 1 \end{pmatrix}$. Множество пометок на узлах есть множество характеризованных

обобщенных лексем: $W^R = W^{RL} \cup W^{LF} \cup W^{ID} \cup W^{FL}$, где W^{RL} – реальные лексемы языка, W^{LF} – символьные обозначения лексических функций. $W^{FL} = \{Q\}$, Q есть символ фиктивной (пустой) лексемы, которая служит для обозначения узла, не получающего "вещественного" означающего в реальной фразе, но тем не менее его присутствие в дереве глубинного синтаксиса продиктовано семантическими соображениями (пример – незаполненная смысловая валентность у глагола).

Модель СЭ на основе грамматик вида (1.2), исследование ее свойств в аспекте проблем алгоритмической разрешимости и вычислительной сложности детально обсуждается в [112]. Указанная модель использует разнообразную информацию о каждом слове ЕЯ в виде словоизменительных, словообразовательных, синтаксических семантических и стилистических характеристик слова, описываемых в Толково-Комбинаторном Словаре (ТКС) [118]. В частности, синтаксические и семантические характеристики используются при описании условий применимости правил множества Π^R.

Актуальными здесь являются проблемы автоматизации накопления и систематизации знаний, представляемых ТКС, непосредственно на основе текстовых массивов.

1.4. Анализ формальных понятий как инструмент концептуальной кластеризации

Как отмечал И.А. Мельчук в [45, стр. 18-20], в модели языка как преобразователя "Смысл⇔Текст" следует выделить лингвистическую (декларативную) часть, которая представляет собой множество правил соответствия между смыслами и ЕЯ-текстами, и алгоритмическую

(процедурную) часть, реализующую механизм использования указанных соответствий. Причем предполагаются переходы от сложных (получаемых операциями комбинирования) смыслов к столь же сложным текстам (то есть также получаемых посредством комбинирования) и наоборот.

Сказанное говорит о том, что указанная составляющая модели “Смысл⇔Текст” должна быть динамической. На практике это означает следующее: будучи независимые от конкретной процедуры реализации, правила соответствия между смыслами и текстами предполагают конкретизацию условий их применения на ЕЯ-текстах заданной предметной области.

Как отмечал академик Ю.Д.Апресян [3, стр. 335-336], ограничения, накладываемые, в частности на синонимические преобразования глубинных синтаксических структур, зависят как от особенностей отдельных слов, так и целых пластов лексики. Актуальным здесь является выбор подходящей модели представления знаний о синонимии, в совокупности с подходом к их систематизации и упорядочиванию.

Согласно формулировке *Задачи 1.2*, а также данному в [45, стр. 151] определению условия применимости правила ЛФ-синонимического преобразования, прецедент класса СЭ определяется в первую очередь совокупностью требований к синтаксическим и семантическим свойствам тех лексических единиц, которые участвуют в выполняемой посредством правила замене. При этом информация, связанная с лексемой, включает денотативный и смысловой компоненты.

Определение 1.5. Денотат ЕЯ-слова есть множество сущностей реального мира, которые этим словом могут быть правильно названы.

В отличие от референции денотат является частью значения слова и не зависит от контекста конкретной ситуации употребления ЕЯ.

Понятие смысла слова в целом сходно с понятием смысла высказывания даваемым *Определением 1.3*.

Определение 1.6. Смысл слова определяется как множество отношений вида “денотат-денотат” (именуемых также смысловыми отношениями), существующих между данным словом и другими словами в заданном естественном языке.

В логике различие между смыслом и денотатом определяется с помощью экстенсионала и интенсионала.

Определение 1.7. Экстенсионал (объем понятия) есть класс сущностей, именуемых заданным словом.

Определение 1.8. Интенсионал (содержание понятия) есть множество признаков, определяющих класс сущностей из экстенсионала.

Как следует из *Определений 1.7* и *1.8*, экстенсионал соответствует денотату, интенсионал – смыслу слова.

Само описание слова с точки зрения объема и содержания понятия, обозначаемого словом, составляет основу кластеризации слов, наиболее естественно реализуемой с применением методов Анализа Формальных Понятий (АФП) [115].

Определение 1.9. АФП – это метод анализа данных, основанный на математической теории решеток [4]. Основой АФП является доказанная Г. Биркгофом теорема [4] о том, что для любого бинарного отношения можно построить полную решетку.

При использовании данного метода некоторая исследуемая область знаний описывается в терминах набора объектов и признаков (атрибутов), затем вводится описание формального контекста. Далее для заданного контекста формируется множество формальных

понятий и строится решетка, которая может быть визуально отображена диаграммой линий. Формализация понятий и их последующий анализ с помощью решетки позволяют оперировать данными на семантическом уровне без потери или недопустимого упрощения объектов и их признаков.

Классификация объектов и результаты анализа данных с помощью АФП могут быть интерпретированы исследователем для заданной предметной области.

Приведем используемые далее основные определения из теории АФП.

Пусть G – множество объектов, M – множество признаков для объектов из G. Имеем также бинарное отношение $I \subseteq G \times M$. Если $g \in G$ и $m \in M$, то $g\,\mathrm{Im}$ имеет место тогда и только тогда, когда g обладает признаком m.

Определение 1.10. Тройка $K = (G, M, I)$ называется формальным контекстом. При этом для произвольных $A \subseteq G$ и $B \subseteq M$ вводится пара отображений: $A' = \{m \in M \mid \forall g \in A : g\,\mathrm{Im}\}$ и $B' = \{g \in G \mid \forall m \in B : g\,\mathrm{Im}\}$.

Определение 1.11. Пара множеств (A, B), таких что $A \subseteq G$, $B \subseteq M$ и $A' = B$, $B' = A$, называется Формальным Понятием (ФП) с объемом A и содержанием B.

Определение 1.12. ФП (A_1, B_1) называют подпонятием для ФП (A_2, B_2), если $A_1 \subseteq A_2$. При этом (A_2, B_2) называют суперпонятием для ФП (A_1, B_1) (обозначается как $(A_1, B_1) \le (A_2, B_2)$). Отношение $\le$ будем называть отношением порядка для формальных понятий.

Определение 1.13. Формальные понятия C_1 и C_2 считаются сравнимыми, если либо $C_1 \le C_2$, либо $C_2 \le C_1$. В противном случае эти ФП называют несравнимыми.

Определение 1.14. Множество всех ФП контекста $K = (G, M, I)$ вместе с заданным на нем отношением $\leq$ обозначают $\mathfrak{R}(G, M, I)$ и называют решеткой формальных понятий.

Определение 1.15. Подмножество множества формальных понятий, в котором каждые два элемента являются сравнимыми, называют цепочкой, а если каждые два элемента являются несравнимыми, называют антицепочкой.

Определение 1.16. Под областью в решетке ФП понимается набор формальных понятий, связанных отношением порядка с одним Наибольшим Общим Подпонятием (НОПП) и/или одним Наименьшим Общим Суперпонятием (НОСП). В роли НОПП может выступать наименьшее ФП в решетке, а в роли НОСП – вершинное ФП.

Определение 1.17. ФП C_2 называется соседним по отношению к ФП C_1 в решетке $\mathfrak{R}$, если они имеют НОСП, отличное от вершинного ФП в этой решетке.

Замечание. АФП по определению есть инструмент концептуальной кластеризации, так как $\forall (A, B) \in \mathfrak{R}$ есть класс с заданной интерпретацией в виде содержания – множества B.

Далее в работе мы покажем, каким образом с помощью АФП выполняется формирование и классификация условий применимости ЛФ-синонимических преобразований в *Задаче 1.2*, решается задача текстовой кластеризации, включающая *Задачу 1.1* в качестве подзадачи.

Кроме того, посредством АФП реализуется механизм согласования различных уровней синонимии в естественном языке, и определяются меры схожести ситуаций языкового употребления.

Выводы

Анализируя задачу текстовой кластеризации, описанную нами в общих чертах в разделе 1.3, можно констатировать, что:

– ситуация языкового употребления может служить источником знаний как о лексико-функциональной синонимии, представляющей верхний уровень иерархии знаний о синонимии, так и о произвольных случаях СЭ в ЕЯ;

– АФП представляет собой инструмент формирования и кластеризации понятий, которые будут соответствовать классам СЭ;

– решетка формальных понятий является удобным формализмом для представления текстовой информации в сжатом виде [42], сами тексты при этом объединяются в классы по сходству признаков сочетаемости слов относительно контекстов, определяемых ситуациями языкового употребления.

При этом задача иерархизации знаний о синонимии в заданном ЕЯ сводится к совокупности следующих подзадач:

– выделение и кластеризация множества отношений между объектами-фигурантами ситуации вида (1.1);

– формирование прецедентов для ситуаций ЛФ-синонимии в соответствии со сформулированной нами *Задачей 1.2* на основе полученных отношений;

– определение мер схожести для ситуаций языкового употребления на основе формализованного представления знаний о синонимии в заданном предметно-ориентированном подмножестве естественного языка.

Язык глубинных синтаксических структур как средство описания синтаксических и лексических правил синонимического перифразирования при всех своих несомненных достоинствах

обладает и одним существенным недостатком, а именно: на указанном уровне текст представляется пофразно, каждая фраза соответствует простому распространенному предложению. Отсюда возникает проблема полноты описания смысла при формировании прецедентов классов СЭ для *Задачи 1.2*. Если одна из форм описания рассматриваемой ситуации действительности представлена ЕЯ-высказыванием, состоящим более чем из одной фразы, то в соответствии с *Задачей 1.1*, выделение множеств ситуаций, описываемых текстом, и объектов, значимых в этих ситуациях, должно производиться только на основе анализа сходств классов СЭ всех фраз, составляющих высказывание.

Решению задачи построения единого семантического образа ЕЯ-высказывания на уровне глубинного синтаксиса посвящается вторая глава работы.

Глава 2

СЖАТИЕ СМЫСЛОВОЙ ИНФОРМАЦИИ НА УРОВНЕ ГЛУБИННОГО СИНТАКСИСА

На основе полученного в первой главе формализованного описания процесса установления СЭ в настоящей главе исследуется проблема полноты представления смысла при описании синонимического варьирования абстрактной лексикой на уровне глубинного синтаксиса. Ставится и решается задача построения формальных семантических образов сверхфразовых единств на указанном языковом уровне в высказываниях более чем из одного простого распространенного предложения русского языка.

2.1. Концептуальная модель процесса распознавания смысловой взаимной дополняемости фраз в сравниваемых по смыслу высказываниях естественного языка

Целью настоящего раздела является описание на функциональном уровне задачи увеличения полноты описания смысла в формальном образе ЕЯ-текста при установлении его эквивалентности смысловому эталону-образцу с использованием определяемых в этом же разделе основополагающих понятий и терминологии.

Прежде чем описывать процесс построения формальных семантических образов сверхфразовых единств, введем ряд понятий, характеризующих рассмотренный в [112] процесс установления семантической эквивалентности текстов в рамках подхода “Смысл⇔Текст”. Говоря о сравниваемых текстах, будем считать, что

в общем случае сравниваемые ЕЯ-тексты состоят из различного числа фраз, а одна фраза соответствует простому распространенному ЕЯ-предложению.

Во-первых, предложенная в [112] модель семантической эквивалентности работает с совокупностями деревьев глубинно-синтаксических структур фраз к каждому из этих деревьев должно быть применено одно или несколько правил синонимического перифразирования. В [112] исследуется алгоритмическая сложность задачи применения правила к помеченному дереву без рассмотрения анализируемых входом правила компонент этого дерева. Подобное рассмотрение правил синонимических преобразований не позволяет говорить о необходимых и достаточных признаках синонимии двух фраз по анализу применимости к ним правил синонимических преобразований и, как следствие, целесообразности синонимических трансформаций того или иного типа, что позволило бы в значительной степени сократить перебор в задаче установления семантической эквивалентности ЕЯ-текстов. Поэтому рассмотрим более подробно процесс применения правила расширенной лексико-синтаксической Δ-грамматики к некоторому дереву с пометками на ветвях и в узлах, в содержательной интерпретации – лексического правила с обслуживающим его синтаксическим правилом синонимических замен к дереву глубинного синтаксиса.

Согласно принятому в теории языка как преобразователя “Смысл⇔Текст” делению правил синонимических преобразований на лексические и синтаксические и взаимозависимостью применений правил указанных типов, в процессе применения правила следует выделить:

- определение поддерева, заменяемого лексическим правилом+первым из обслуживающих его синтаксических правил с фиксацией номеров правил;
- определение ключевого слова (C_0) комплекса лексических единиц, заменяемых лексическим правилом ([45, стр. 150]).

Определение 2.1. Лексической Синонимической Конструкцией (ЛСК) будем называть комплекс лексических единиц (обобщенных лексем и их лексических коррелятов) и связывающих их глубинно-синтаксических отношений, замена которого описывается некоторым лексическим правилом синонимического перифразирования. Каждой ЛСК соответствует свое ключевое слово C_0, либо непосредственно входящее в нее, либо выраженное в значениях ЛФ от C_0 (лексических коррелятов C_0) в комплексе составляющих ЛСК лексических единиц.

Как следует из *Определения 2.1*, каждой ЛСК соответствует свое ключевое слово C_0, являющееся ключевым словом заменяемого лексическим правилом комплекса лексических единиц и входящее в ЛСК либо непосредственно, либо выражаемое в значениях ЛФ от C_0.

На основе полученного определения ЛСК сформулируем необходимые и достаточные условия ЛФ-синонимии глубинных синтаксических структур. Согласно определению, сравниваемым ГСС здесь соответствуют эквивалентные СемП, выраженные одним и тем же семантическим графом. Причем элементы этого СГ группируются в разных ГСС в одинаковые пучки, выраженные разными лексическими единицами, соотношения между которыми описываются с помощью аппарата стандартных ЛФ.

Определение 2.2 (необходимое условие ЛФ-синонимии ГСС). Будем считать, что ГСС фраз F_1 и F_2: $F_1 \neq F_2$ удовлетворяют

необходимому, но не достаточному условию ЛФ-синонимии, если их ЛСК относится к одному и тому же ключевому слову C_0.

Данное условие позволяет определить возможность наличия некоторой ГСС в множестве деревьев, получаемом для заданной ГСС с применением лексических правил синонимических замен без построения этого множества. Действительно, если для рассматриваемых ГСС их ЛСК относятся к разным ключевым словам, то получение на основе некоторой ГСС дерева глубинного синтаксиса, эквивалентного заданному, возможно только посредством чисто синтаксических замен, не требующих замен лексики.

Замечание. Поскольку лексические замены ведутся относительно определенного ключевого слова C_0, то невыполнение необходимого условия ЛФ-синонимии для одних ЛСК в сравниваемых деревьях не означает невозможности отношения ЛФ-синонимии между рассматриваемыми деревьями, поскольку к одному и тому же дереву глубинного синтаксиса может быть применено несколько лексических правил перифразирования, что позволяет говорить об относительности ЛФ-синонимии.

Представим вход правила ЛФ-синонимической замены как описание поддерева, заменяемого первым из обслуживающих данное лексическое преобразование синтаксических правил, внутри которого содержится описание поддерева, заменяемого лексическим правилом. Тогда определение возможности применения синонимических преобразований из заданного множества Π^R есть определение применимости каждого правила $\pi_i \in \Pi^R$, с выделением ключевого слова ЛСК и представлением результата в виде списка пар:

$$\left\{\left(\pi_i, C_0(i)\right) : i = 1,\ldots,\left|\Pi^R\right|\right\}. \qquad (2.1)$$

Если для некоторой ГСС, входящей в множество ГСС смыслового описания одного высказывания, ключевое слово C_0 одного из элементов списка (2.1) совпадает с ключевым словом одного из элементов аналогичного списка у некоторой ГСС из смыслового описания второго, “эталонного” высказывания, то дальнейшие действия по установлению эквивалентности указанных ГСС включают в себя:

- построение некоторой последовательности лексических преобразований, приводящих поддеревья исходных ГСС, заменяемые лексическими правилами + первыми из обслуживающих их синтаксических правил, к виду с одинаковой ЛСК;
- сравнение путем наложения, начиная с вершины, при совмещении одноименных стрелок преобразованных ГСС на предмет эквивалентности.

Определим понятие эквивалентности (равенства) ГСС.

Определение 2.3. Помеченные деревья T_1 и T_2 (в содержательной интерпретации – деревья глубинного синтаксиса) являются эквивалентными (равными, тождественными), если они изоморфны [7] таким образом, что для всякого узла α дерева T_1 его образ $f(\alpha)$ в дереве T_2 имеет одинаковую с ним пометку.

Как показано в [7], применение некоторого преобразования в Δ-грамматике сводится к последовательному выполнению:

- декомпозиции [15] исходного дерева с выделением заменяемого поддерева и расстановкой композиционных меток, обозначающих выделенные узлы;
- композиции [7] дерева верхнего контекста заменяемого дерева, заменяющего дерева и деревьев нижнего контекста в

соответствии с порядком, задаваемым композиционными метками.

Таким образом, для установления эквивалентности деревьев глубинного синтаксиса T_1 и T_2, приведенных к виду с одинаковой ЛСК, необходимо вначале выполнить сравнение замененных поддеревьев, включающих ЛСК, а затем – деревьев верхнего и нижнего контекста замененных поддеревьев. Последние в результате последовательности трансформаций остаются без изменений.

Показанное свойство ЛФ-синонимических преобразований позволяет рассматривать ЛФ-синонимию ГСС при задании их ЛСК относительно одного и того же ключевого слова C_0 как частный случай семантического повтора на основе значений лексических функций самостоятельных лексем [78]. При этом ЛСК рассматриваются в качестве элементов повтора, представляя собой комбинаций значений лексических функций заданного ключевого слова, связанных отношениями глубинного синтаксиса.

Определим формально взаимную дополняемость глубинных синтаксических представлений при задании их ЛСК относительно одного и того же ключевого слова.

Определение 2.4. Будем считать, что отвечающие необходимому условию ЛФ-синонимии (в соответствии с *Определением 2.2*) деревья T_1 и T_2 удовлетворяют необходимому (но не достаточному!) условию смысловой взаимной дополняемости, если существует последовательность ЛФ-синонимических преобразований, приводящих T_1 и T_2 к виду с одинаковой ЛСК.

Для дальнейшего изложения введем в рассмотрение семантические словоизменительные характеристики лексем, представляемых в узлах дерева глубинного синтаксиса. Согласно

данному в [45, стр. 144] определению, к таковым относятся число для существительных и вид, время, наклонение – для глагола.

Определение 2.5. Будем считать, что удовлетворяющие (согласно *Определению 2.4*) необходимому условию взаимной дополняемости и приведенные к виду с одинаковой ЛСК деревья T_1 и T_2 взаимно дополняют друг друга, если они изоморфны так, что для всякого узла α дерева T_1 его образ $f(\alpha)$ в дереве T_2:

- либо содержит информацию об одной и той же характеризованной обобщенной лексеме ([45, стр.144]) данного ЕЯ, не являющейся нулевой (фиктивной) лексемой ([45, стр.143]);
- либо представляет обозначенную символом Q фиктивную лексему с теми же семантическими словоизменительными характеристиками, что и ненулевая характеризованная обобщенная лексема, информация о которой содержится в узле α;
- либо представляет ненулевую характеризованную обобщенную лексему с теми же семантическими словоизменительными характеристиками, что и фиктивная лексема, информация о которой содержится в узле α.

Следствие. Рассматриваемая *Определением 2.3* эквивалентность (равенство) ГСС является частным случаем взаимной дополняемости деревьев глубинного синтаксиса.

Замечание. В реальных ЕЯ-текстах достаточно много случаев, когда удовлетворяющие (согласно *Определению 2.4*) необходимому условию взаимной дополняемости и приведенные к одинаковой ЛСК деревья T_1 и T_2 не могут взаимно дополнять друг друга. Причина кроется в том, что существует как минимум один узел α дерева T_1,

образ $f(\alpha)$ которого в дереве T_2 содержит информацию о ненулевой характеризованной обобщенной лексеме с теми же семантическими словоизменительными характеристиками, что и отличная от нее ненулевая характеризованная обобщенная лексема, представляемая узлом α. Будем считать, что в этом случае T_1 и T_2 имеют *ложную взаимную дополняемость*.

Таким образом, увеличения полноты смыслового описания текста, сравниваемого с эталоном на уровне глубинного синтаксиса, можно достичь суммированием глубинных синтаксических структур, взаимно дополняющих друг друга, путем наложения при совмещении одноименных стрелок с “заполнением мест”, соответствующих фиктивным (нулевым) лексемам. При этом исходные ГСС, сведенные к единой (“суммарной”) ГСС, исключаются из смыслового описания анализируемого текста. В содержательной лингвистической интерпретации это означает для анализируемого текста построение образов сверхфразовых единств [81] на уровне глубинного синтаксиса.

Определение 2.6. Формальным образом сверхфразового единства (в дальнейшем – сверхфразовым единством) на глубинном синтаксическом уровне представления смысловых образов фраз будем называть дерево глубинного синтаксиса, полученное суммированием глубинных синтаксических структур, взаимно дополняющих друг друга по *Определению 2.5*, путем наложения при совмещении одноименных стрелок с “заполнением мест”, соответствующих фиктивным (нулевым) лексемам.

С учетом введенного в модель семантической эквивалентности распознавания сверхфразовых единств, функционирование механизма установления семантической эквивалентности высказываний будет

представляться следующей концептуальной моделью, полученной расширением соответствующей модели, представленной в [108,112].

Для заданного ЕЯ Y вводится в рассмотрение язык смыслов Y_S. Как было показано нами в разделе 1.3, при рассмотрении смысла как инварианта синонимических преобразований в качестве Y_S будет выступать язык глубинного синтаксиса. Сам язык Y_S при этом представляется упорядоченной пятеркой:

$$Y_S = \langle L_S, \Gamma_S, \Pi_S, Q_S, U_S \rangle, \qquad (2.2)$$

где L_S – лексика языка Y_S;

Γ_S – синтаксис языка Y_S;

Π_S – процедура установления соответствий между фразами языков Y и Y_S;

Q_S – процедура, с помощью которой решается проблема эквивалентности в языке Y_S;

U_S – процедура, преобразующая смысловое представление анализируемого текста на основе учета описанных выше семантических повторов.

Процедура Q_S содержит допустимые L_S и Γ_S лексические и синтаксические правила преобразований эквивалентных смысловых образов друг в друга (фактически – правила ЛФ-синонимических преобразований ГСС). Компонента U_S описывает приведение фраз, связанных по смыслу в языке Y_S (по мнению носителя языка Y), к формальному представлению, допускающему нахождение искомого суммарного смысла (в содержательной интерпретации – к виду с одинаковой ЛСК). Кроме того, в составе U_S содержатся правила

построения единого смыслового образа для “приведенных” фраз языка Y_S. Исходя из вышесказанного, представим компоненту U_S упорядоченной двойкой:

$$U_S = \langle Q_U, S_U \rangle, \qquad (2.3)$$

где Q_U – процедура приведения фраз в Y_S, связанных по смыслу (по мнению носителя языка Y), к формальному представлению, допускающему нахождение искомого суммарного смысла (т.е. к виду с одинаковой ЛСК). Процедура Q_U использует допустимые L_S и Γ_S лексические и синтаксические преобразования с наложением необходимых ограничений;

S_U – процедура, содержащая правила построения единого смыслового образа для "приведенных" фраз из Y_S (суммарного смысла в языке Y_S).

Язык Y_S обладает следующими свойствами, актуальными для решения задачи распознавания семантических повторов в сравниваемом с эталоном ЕЯ-тексте:

1) если фразы F_1 и F_2 языка Y (по мнению его носителя) эквивалентны по смыслу, то с помощью Π_S обе эти фразы либо переводятся в одну и ту же фразу языка Y_S, либо переводятся в две фразы Φ_1 и Φ_2, но такие, что Φ_1 и Φ_2 эквивалентны в Y_S;
2) если фразы F_1 и F_2 языка Y (по мнению его носителя) взаимно дополняют друг друга по смыслу, то полученные с помощью процедуры Π_S образы Φ_1 и Φ_2 этих фраз в языке Y_S процедурой Q_U сводятся к виду, допускающему нахождение искомого суммарного смысла, а затем посредством процедуры S_U

переводятся в одну фразу Φ языка Y_S, соответствующую образу суммарного смысла;

3) если фразам F_1 и F_2 языка Y соответствуют полученные с помощью процедуры Π_S фразы Φ_1 и Φ_2 языка Y_S, сводимые процедурой Q_U к представлению, допускающему нахождение искомого суммарного смысла, но не сводимые с помощью процедуры S_U в единую фразу языка Y_S, то фразы F_1 и F_2 следует считать фразами с ложной смысловой взаимной дополняемостью.

Кроме того, предполагается наличие необходимых, но не достаточных признаков наличия семантической связи между фразами из Y на основе анализа их образов в Y_S (см. *Определения 2.2* и *2.4*). В силу родственной природы задач установления семантической эквивалентности и распознавания семантических повторов указанные признаки берутся в качестве необходимых, но не достаточных признаков эквивалентности фраз. Для анализа возможностей использования таких признаков рассмотрим более подробно структуру множества фраз постулируемого языка смыслов Y_S.

Следует отметить, что введение в рассмотрение смысловых повторов вводит в модель двойное разбиение множества Φ_S фраз языка Y_S. С одной стороны, указанное множество разбивается на непересекающиеся подмножества:

$$\Phi_S = \Phi_{S1} \cup \Phi_{S2} \cup \ldots \cup \Phi_{Sk}, \qquad (2.4)$$

в каждом из которых фразы эквивалентны между собой, но ни одна фраза $\Phi_1 \in \Phi_{Si}$ для $i = 1, \ldots, k$ не будет эквивалентна ни одной другой фразе $\Phi_2 \in \Phi_{Sj}$ для $j = 1, \ldots, k$, если $i \neq j$.

С другой стороны, то же самое множество фраз можно разбить на непересекающиеся множества Φ_{LSCi}, имеющих ЛСК, задаваемые каждое относительно своего ключевого слова. При этом особым подмножеством множества Φ_S будет множество Φ_{SYNT} фраз языка Y_S, для которых не может быть определена ЛСК (могут быть применены только синтаксические трансформации, допустимые Γ_S):

$$\Phi_S = \Phi_{LSC1} \cup \Phi_{LSC2} \cup \ldots \cup \Phi_{LSCl} \cup \Phi_{SYNT}. \qquad (2.5)$$

При этом Φ_{Si}, Φ_{LSCj} и Φ_{SYNT} связаны друг с другом следующим образом. Каждое из Φ_{Si} может включать в себя элементы разных Φ_{LSCj}, а также элементы Φ_{SYNT}. Иначе говоря, каждое Φ_{Si} есть множество, которое может включать подмножества нескольких множеств Φ_{LSCj} плюс некоторое подмножество множества Φ_{SYNT}. Содержательно это соответствует принципу относительности выделения ЛСК: к одной и той же фразе Φ могут быть применены несколько правил преобразований, причем каждое относительно своей ЛСК. Более того, что особенно важно для построения процедуры S_U, в каждом множестве Φ_{LSCj} выделяется два подмножества:

– множество пар фраз, взаимно дополняющих друг друга по смыслу:

$$\Phi_{Sj}^{T} = \{(\Phi_1, \Phi_2) : U_S(\Phi_1, \Phi_2, \Phi) = true\} \subset \Phi_{LSCj} \times \Phi_{LSCj},$$

подмножеством которого является множество пар фраз, эквивалентных между собой;

– множество пар фраз с ложной взаимной дополняемостью.

Можно показать, что если $(\Phi_1, \Phi_2) \in \Phi_{Sj}^T$ и $(\Phi_3, \Phi_4) \in \Phi_{Sj}^T$, то из этого не следует, что указанное соотношение будет справедливым и для пар (Φ_1, Φ_3), (Φ_2, Φ_4), (Φ_1, Φ_4), (Φ_2, Φ_3).

Таким образом, модель (2.2) отвечает выдвинутому в *Главе 1* требованию согласования различных уровней синонимии между собой. Тем не менее, предложенная модель является концептуальной, не имея в своем составе средств описания модели и аппарата манипулирования данными в плане:

- описания механизма применения определенных в процедуре Q_S лексических и синтаксических преобразований фраз множества Φ_S;
- описания процедуры U_S;
- описания взаимодействия процедуры Q_S с процедурой U_S в процессе установления эквивалентности в языке Y_S.

Указанные задачи предполагают построение и исследование модели процесса приведения глубинной синтаксической структуры к некоторому заданному виду. Такая модель ориентирована на формализованное описание входа/выхода правила как информационного элемента и предусматривает различные ситуации его активизации. Этим вопросам посвящаются три последующие раздела настоящей главы.

2.2. Построение системы целевых выводов в Δ-грамматике

В данном разделе решается задача приведения глубинных синтаксических структур фраз к виду, допускающему нахождение суммарного смысла (к виду с одинаковой ЛСК). Рассматривается

построение системы целевых выводов в Δ-грамматике, реализуемое процедурой Q_U в составе концептуальной модели (2.2)–(2.3).

Решение задачи получения на основе исходной ГСС другой ГСС, удовлетворяющей некоторым функциональным требованиям, при использовании заданной системы правил синонимического преобразования помеченных деревьев, требует исследования динамики функционирования Δ-грамматики, которая моделирует указанную систему. С этой целью в настоящем разделе мы рассмотрим логическую модель отдельного правила Δ-грамматики для последующего описания структуры информационного пространства, соответствующего системе таких правил.

В настоящей работе, говоря о правилах Δ-грамматики, мы имеем в виду подмножество произвольных элементарных преобразований [7,112], которыми моделируются глубинные синтаксические преобразования конкретного рассматриваемого ЕЯ. При дальнейшем изложении, говоря о правилах Δ-грамматики, мы будем подразумевать произвольные элементарные преобразования, опуская этот термин.

Сформулируем задачу достижимости ЛСК с заданными свойствами следующим образом. Пусть имеется дерево ГСС T_1^{π}, удовлетворяющее требованиям входа некоторого правила $\pi \in \Pi^R$ Δ-грамматики Γ^{RL}. Будем рассматривать соответствующие правилам синонимических преобразований переходы от одной ГСС к другой, ЛФ-синонимичной с ней как односторонние, а если некоторое правило выполняется в обе стороны, то ему будут соответствовать два возможных перехода, каждый из них выполняется в своем направлении. Следует отметить, что в отличие от динамических информационных структур, используемых для построения интерактивных графических систем [80,105,106], связи между

входами и выходами правил как информационными элементами задаются изначально и не могут быть изменены в процессе функционирования системы.

Рассмотрим работу некоторого правила $\pi \in \Pi^R$. В общем случае здесь следует выделить:

- состояние, соответствующее заменяемому дереву T_1^π;
- состояние, соответствующее заменяющему дереву T_2^π;
- условие $r(\pi)$ срабатывания правила π для T_1^π и T_2^π.

Иными словами, мы имеем простейший случай задачи достижимости ЛСК с заданными свойствами на информационном пространстве, заданном входами и выходами правил $\pi \in \Pi^R$. Решение такой задачи есть ответ на ряд вопросов, а именно:

- удовлетворяет ли исходное дерево требованиям входного дерева T_1^π рассматриваемого правила π;
- удовлетворяет ли целевое дерево требованиям выходного дерева T_2^π правила π;
- возможен ли переход от T_1^π к T_2^π с учетом информационного наполнения исходного и целевого деревьев в совокупности с характером условия $r(\pi)$.

Более общий случай задачи достижимости ЛСК с заданными свойствами отличается от описанного простейшего тем, что:

- рассматриваются входы и выходы не одного, а разных правил $\pi_1 \in \Pi^R$ и $\pi_2 \in \Pi^R$, $\pi_1 \neq \pi_2$;
- исследуется возможность не одного, а последовательности переходов от T_1^π к T_2^π.

Условие $r(\pi)$ применимости правила $\pi \in \Pi^R$ содержит список требований к элементам T_1^π и T_2^π, представляя собой формальное описание допустимости перехода из состояния T_1^π в состояние T_2^π. Учитывая особенности реальных систем перифразирования, наиболее целесообразным для каждого $\pi \in \Pi^R$ рассматривать множество R_π условий применимости, из которых для срабатывания правила должно выполниться как минимум одно $r_i(\pi) \in R_\pi$.

Переход из состояния T_1^π в состояние T_2^π возможен при условии свершения системы событий, соответствующих обнаружению в глубинном синтаксическом дереве на определенных позициях узлов с заданными характеристиками. В отличие от динамических информационных структур, используемых для построения интерактивных графических систем, в рассматриваемой задаче изменение состояния системы может быть вызвано не только отдельным событием, но и их системой, причем в большинстве правил имеет место именно система событий.

Опишем формально совокупности событий, определение которых используется компонентой R_π. С учетом вышесказанного, множество R_π есть множество систем событий из множества X всех событий, допустимых всеми системами правил множества Π^R. В случае одного события, ведущего к изменению состояния некоторой заданной системы правил, рассматривается система из одного события как частный случай системы из *n* событий. В содержательной интерпретации каждое $x^i \in X$ есть обнаружение в глубинном синтаксическом дереве на определенной позиции узла с некоторыми

характеристиками и ему соответствует значение либо *“true”* (обнаружение), либо *“false”* (не обнаружение). Каждая система событий представляется списком $\left(x^1, x^2, \ldots, x^n\right)$. Применение правила $\pi \in \Pi^R$ рассматривается как переход из состояния T_1^{π} в состояние T_2^{π}, который будет возможен, если существует система событий $\left(x^1, x^2, \ldots, x^n\right)$ такая, что $x^1 \wedge x^2 \wedge \ldots \wedge x^n = true$ и существует условие применимости $r_j(\pi) \in R_{\pi}$, описываемое логической формулой:

$$r_j(\pi) = x^1 \wedge x^2 \wedge \ldots \wedge x^n. \qquad (2.6)$$

Теперь мы можем дать формальное определение компонента R_{π}, отвечающего за применимость правила $\pi \in \Pi^R$. Правило π может быть применено к дереву T_1^{π}, если выполняется одно из условий $r_j(\pi) \in R_{\pi}$: $\vee_{j=1}^{m} r_j(\pi) = true$, где $m = \left|R_{\pi}\right|$. Обозначим $\vee_{j=1}^{m} r_j(\pi)$ для дальнейшего использования как r_{12}. Условие r_{12} следует интерпретировать как “определение события, разрешающего переход от T_1^{π} к T_2^{π}”.

Применение правила $\pi \in \Pi^R$ сводится к выполнению перехода:

$$\pi(r_{12}): T_1^{\pi} \xrightarrow{\pi(r_{12})} T_2^{\pi}. \qquad (2.7)$$

Предложенная модель правила Δ-грамматики естественным образом согласуется с математическим аппаратом сетей Петри [39,70]. Отдельному правилу соответствует элементарная сеть Петри вида

$$N = \{P, T, F, H, M_0\}. \qquad (2.8)$$

При этом множество состояний правила есть множество P позиций (мест) сети: $P = \{p_1, p_2\}$, где $p_1 \Leftrightarrow T_1^{\pi}$, а $p_2 \Leftrightarrow T_2^{\pi}$.

Множество возможных переходов T сети представлено единственным описываемым моделируемым правилом переходом из состояния T_1^{π} в состояние T_2^{π} : $t = \pi(r_{12})$: $p_1 \xrightarrow{t} p_2$. Компоненты F и H представляют отображения, задаваемые матрицами инцидентности $F : P \times T \to \{0,1\}$ и $H : T \times P \to \{0,1\}$, соответственно. Согласно определению [70], для любой $p_i \in P$ $F(p_i, t) = 1$, если p_i является входной позицией перехода t. Аналогично $H(t, p_i) = 1$, если p_i – выходная позиция перехода t. Для сети вида (2.8), моделирующей работу правила Δ-грамматики, имеем: $F(p_1, t) = 1$, $F(p_2, t) = 0$, $H(t, p_1) = 0$, $H(t, p_2) = 1$. Количество допустимых разметок сети, отождествляемых со сценариями [105], здесь равно двум. Каждый сценарий соответствует совокупности активных в текущий момент информационных элементов. В рассматриваемой модели одновременно активным может быть только один элемент, соответствующий либо входу, либо выходу правила. Поскольку множество мест в сети изначально упорядочено (порядок соответствует состояниям моделируемого правила), каждая из допустимых разметок может быть представлена в виде двоичного вектора длины, равной числу позиций, то есть 2. Начальной маркировке соответствует вектор $M_0 = (1,0)$, второй из допустимых маркировок – вектор $M = (0,1)$. Вторая разметка является тупиковой.

Ввиду того, что множество правил Π^R используется программой, а не пользователем-человеком, то следует формально определить функцию активизации входа правила, являющейся функцией активизации (запуска или начальной маркировки [70]) сети Петри. Указанная функция формально определяется как логическая

функция, выдающая либо *“true”*, если анализируемое дерево глубинного синтаксиса функционально соответствует входному дереву T_1^{π} правила $\pi \in \Pi^R$, либо *“false”* в противном случае. По значению этой функции происходит (в случае *“true”*), либо не происходит (в случае *“false”*) начальная маркировка $M_0 = (1,0)$ рассматриваемой сети Петри.

Рассмотрим ограничения, накладываемые на классический аппарат сетей Петри, применительно к моделированию правила Δ-грамматики.

Во-первых, правило $\pi \in \Pi^R$ моделируется элементарной сетью Петри, в которой количество фишек (маркеров) в каждой позиции не превышает 1. Следует отметить, что это ограничение накладывается из содержательных особенностей моделируемого объекта, а не является свойством топологии сети. В содержательном плане это означает, что за один проход (одно срабатывание правила Δ-грамматики) не может быть обработано более одного дерева.

Во-вторых, введена функция, определяющая возможность срабатывания перехода при выполнении определенного в классическом аппарате сетей Петри условия срабатывания (наличие фишек в каждой из входных позиций) путем анализа системы событий, сопутствующей активизации позиции, инцидентной данному переходу. Содержательно такое ограничение ведет к появлению тупиковых разметок второго рода: условие активизации инцидентных переходу позиций выполнено, но переход сработать не может, поскольку функция $t = \pi(r_{12})$ активизации перехода $p_1 \xrightarrow{t} p_2$, соответствующая условиям применимости рассматриваемого правила $\pi \in \Pi^R$ выдает *“false”*.

Множество представленных элементарными сетями Петри правил Δ-грамматики можно рассматривать как множество исходных объектов-примитивов для построения в терминах ограниченных сетей Петри [39,70] модели системы правил некоторого подмножества множества Π^R рассматриваемой Δ-грамматики с определением структурных взаимосвязей между ними. При этом сама система правил формируется следующим образом: для каждой пары правил $\{\pi_1, \pi_2\} \subset \Pi^R$, $\pi_1 \neq \pi_2$, входящих в систему, обязательным является выполнение следующего условия: либо вход правила π_2 является выходом для π_1, либо наоборот, вход у π_1 является выходом для π_2.

Следует отметить, что для любой Δ-грамматики такие системы могут быть определены изначально и не обладают свойством динамичности : связи входов и выходов правил детерминированы, а роли пользователя-человека в моделях мультимедиа-приложений [105] соответствует машина, работающая по жестко заданной логике, определяемой системой перифразирования рассматриваемого ЕЯ, что исключает фактор случайности из функционирования приложения.

Рассмотрим динамику функционирования совокупности правил из множества Π^R, образующих систему, для случая, когда одновременно активизируются входы у двух различных правил. Подобным образом функционирует система перифразирования ЕЯ при приведении ГСС фраз к целевому виду.

Отметим, что для построения практически значимой модели системы синонимического перифразирования недостаточно простого описания совокупности переходов от одного ЛФ-синонимичного представления к другому. Простое перечисление правил, условий их применения, обслуживающих правил не учитывает:

- преобразования, выполняемые согласно требованиям Моделей Управления (МУ) предикатных слов, указываемым в их словарных статьях;
- возможность определения по заданной системе правил Δ–грамматики выводимости ГСС с заданными свойствами.

Получение дерева с требуемыми свойствами при распознавании семантических повторов на уровне глубинного синтаксиса означает поиск по совокупности правил заданной Δ–грамматики (с учетом приоритета каждого правила) двух различных выводов, приводящих исходные деревья к представлению, имеющему некоторую заранее заданную общность признаков (в частности, одинаковую ЛСК).

Рассмотрим требования, которым должна удовлетворять модель информационного пространства правил Δ-грамматики в целях адекватности рассматриваемой задаче распознавания семантических повторов.

Во-первых, модель должна описывать взаимосвязи между входными и выходными деревьями различных правил.

Во-вторых, модель должна по заданному дереву, функционально соответствующему входу некоторого правила $\pi \in \Pi^R$, указать деревья, достижимые из заданного применением последовательности правил с максимальной длиной, равной мощности рассматриваемой системы правил, и описать последовательность переходов, соответствующих указанным выводам в Δ-грамматике (последовательность применения правил).

В-третьих, модель должна для двух заданных деревьев T_1^{π} и T_2^{π}, функционально соответствующим входам-выходам некоторых правил множества Π^R, определить возможность приведения к виду с

заданной общностью признаков и указать последовательность выполняемых преобразований.

На основе выдвинутых требований построим формальную концептуальную модель системы синонимического перифразирования глубинных синтаксических структур, формально описываемой в терминах расширенной лексико-синтаксической Δ-грамматики (1.2). Следует особо подчеркнуть, что объектом моделирования здесь являются структурные взаимосвязи между правилами, которые не могут быть описаны в рамках Δ-грамматик, поскольку последние описывают трансформации в языке с точки зрения отдельных преобразований.

Рассмотрим множество T^{Γ} входов и выходов правил $\pi \in \Pi^{R}$, составляющих систему, в качестве множества объектов информационного пространства (множества информационных элементов) заданной Δ-грамматики. При этом T^{Γ} есть объединение множества входов T_1^{Γ} и множества выходов T_2^{Γ}, а модель совокупности правил системы есть совокупность сетей Петри, построенных из моделей отдельных правил как примитивов.

В соответствии с описанием (2.8), для сети N_i одной отдельно взятой i-й системы правил множество позиций P_i включает те элементы множества T^{Γ}, которые соответствуют входам и выходам правил, способных образовывать систему. Множество возможных переходов T_i сети составляют переходы между состояниями, соответствующими входным и выходным деревьям правил. Исходя из содержательной особенности системы правил Δ-грамматики, количество позиций сети N_i, инцидентных переходу, не превышает 1.

Это ограничение не является свойством топологии сети, а естественным образом вытекает из ограничений, накладываемых на примитивы. Мощность множества переходов при этом не превышает величины $\frac{|P_i|!}{(|P_i|-2)!}$. Следует также подчеркнуть следующую особенность матриц инцидентности: $\sum_{k=1}^{|P_i|} F_{kj} = 1$ для $\forall j = 1,\ldots,|T_i|$ и $\sum_{j=1}^{|T_i|} H_{kj} = 1$ для $\forall k = 1,\ldots,|P_i|$. В содержательном плане это означает, что одно и то же правило Δ-грамматики не может описывать генерацию двух различных деревьев, в то же время к одному и тому же дереву может быть применено несколько правил заданной системы.

Теорема 2.1. Пусть N_i – сеть Петри, построенная из примитивов, каждый из которых моделирует работу правила из некоторого подмножества правил заданной Δ-грамматики, образующих систему. Тогда сеть N_i является безопасной в течение всего времени функционирования системы.

Доказательство следует из наложенного на структуру сети ограничения относительно количества позиций, инцидентных переходу (для $\forall t_i^j \in T_i \quad \sum_{k=1}^{|P_i|} H_{jk} = 1$), а также *k*-ограниченности примитива ($k = 1$, так как количество фишек, помечающих позицию сети-примитива не превышает 1).

Активизация (установка начальной разметки M_{0i}) сети Петри рассматриваемой *i*-й системы правил соответствует активизации позиции для входа/выхода того правила, которому функционально соответствует входное дерево. Начальная маркировка или разметка,

как и любая другая из допустимых в рассматриваемой сети разметок, характеризуется тем, что:

- количество маркеров (фишек) в одной позиции не превышает 1, $M_{0i} : P_i \to \{0,1\}$;
- одновременно активизированными могут быть не более одной позиции.

Сеть N_i обладает рядом свойств, касающихся переходов от разметки к разметке.

Во-первых, любая из допустимых для сети разметок может выступать в роли начальной, поскольку изначальная активизация той или иной позиции зависит от того, входу или выходу какого правила системы функционально соответствует входное дерево.

Во-вторых, не исключается наличие тупиковых разметок, обусловленное особенностями моделируемых систем правил. Описываемые Δ-грамматиками реальные системы правил могут включать односторонние преобразования. Для русского языка примером могут служить смысловые импликации [45, стр. 158-159].

В-третьих, начальная разметка может оказаться тупиковой. Этому соответствует ситуация, когда входное дерево функционально соответствует выходу одностороннего преобразования.

Последовательность применяемых правил моделируется последовательностью $\tau = \left(t_i^1, t_i^2, \ldots, t_i^k\right)$ срабатываний переходов:

$$T_1^{\pi} \xrightarrow{\pi_1(r_{12})} T_2^{\pi} \xrightarrow{\pi_2(r_{23})} T_3^{\pi} \to \ldots \to T_k^{\pi} \xrightarrow{\pi_k\left(r_{k\,k+1}\right)} T_{k+1}^{\pi}, \qquad (2.9)$$

где $t_i^1 \Leftrightarrow \pi_1(r_{12})$, $t_i^2 \Leftrightarrow \pi_2(r_{23})$, … , $t_i^k \Leftrightarrow \pi_k\left(r_{k\,k+1}\right)$. При этом происходит последовательная смена разметок:

$$M_{0i} \xrightarrow{t_i^1} M_i^1 \xrightarrow{t_i^2} M_i^2 \to \ldots \to M_i^{k-1} \xrightarrow{t_i^k} M_i^k, \qquad (2.10)$$

где $M_{0i} \Leftrightarrow T_1^{\pi}$, $M_i^1 \Leftrightarrow T_2^{\pi}$, … , $M_i^{k-1} \Leftrightarrow T_k^{\pi}$, $M_i^k \Leftrightarrow T_{k+1}^{\pi}$.

При этом множество достижимости $R(N_i)$ сети N_i находится в зависимости от задания начальной разметки M_{0i}. Если входное дерево функционально соответствует выходу одностороннего преобразования в рассматриваемой Δ-грамматике (тупиковая разметка), то $R(N_i) = \{M_{0i}\}$. Максимальная мощность множества $R(N_i)$ для системы из n_i^{Γ} правил будет иметь место тогда, когда начальная разметка M_{0i} соответствует активизации позиции $p_i^k \in P_i$ с минимальным значением суммы $\sum_{j=1}^{n_i^{\Gamma}} H_{kj}$. Содержательно такая ситуация означает активизацию входа правила системы, в которой все правила двусторонни, либо активизацию входа одностороннего преобразования, который не является выходом никакого другого правила в рассматриваемой системе.

В соответствии с показанным свойством достижимости разметок, множество $L(N_i)$ символьных цепочек, описывающих последовательности срабатывания переходов и составляющих свободный язык рассматриваемой сети Петри, будет определяться в зависимости от задания начальной разметки M_{0i}. Функционирование системы правил Δ-грамматики описывается указанными символьными цепочками. При этом последовательность $t_i^1, t_i^2, \ldots, t_i^{k-1}, t_i^k$ срабатывания переходов есть слово τ в языке $L(N_i)$.

Задача приведения деревьев T_1^{π} и T_{k+1}^{π} к виду с одинаковой ЛСК фактически включает в себя три задачи, связанные с

исследованием свойств сети, построенной из моделей правил как примитивов:

1) определение достижимости разметки M_i^k из начальной разметки M_{0i}. Данная задача есть поиск слова $\tau \in T_i^* \left| M_{0i} \xrightarrow{\tau} M_i^k \right.$, где T_i^* – множество всех слов в алфавите T_i;

2) задача обратимости слова τ: если $\tau \in T_i^* \left| M_{0i} \xrightarrow{\tau} M_i^k \right.$, то существует ли слово $\tau' = \left(t_i^{k'}, t_i^{(k-1)'}, \ldots, t_i^{2'}, t_i^{1'} \right)$:

$$M_{0i} \xleftarrow{t_i^{1'}} M_i^1 \xleftarrow{t_i^{2'}} M_i^2 \leftarrow \ldots \leftarrow M_i^{k-1} \xleftarrow{t_i^{k'}} M_i^k, \qquad (2.11)$$

где $M_{0i} \Leftrightarrow T_1^{\pi}$, $M_i^1 \Leftrightarrow T_2^{\pi}$, … , $M_i^k \Leftrightarrow T_{k+1}^{\pi}$;

3) задача определения оптимального слова $\tau \in T_i^* \left| M_{0i} \xrightarrow{\tau} M_i^k \right.$. Суть: если существуют несколько слов $\tau_1, \tau_2, \ldots, \tau_l$, описывающих последовательности сменяющих друг друга разметок $M_{0i} \xrightarrow{\tau_1} M_i^k$, $M_{0i} \xrightarrow{\tau_2} M_i^k$ и $M_{0i} \xrightarrow{\tau_3} M_i^k$, соответственно, то в качестве оптимального берется обратимое слово минимальной длины.

Отметим, что в отличие от первых двух, третья задача относится к задачам анализа динамики функционирования системы. Действительно, здесь для заданной Δ-грамматики требуется исследовать возможные последовательности срабатываний правил. Решение подобных задач, согласно классической теории сетей Петри, определяется тем, к какому классу языков [39] относится язык $L(N_i)$, порождаемый заданной сетью N_i.

Проведем предварительный анализ языка рассматриваемой сети Петри для отнесения к одному из классов, представленных в [39]. Рассмотрим системные события, соответствующие переходам сети, с точки зрения их тождественности, позволяющей рассматривать одни переходы как “одинаковые”, а другие – как “разные”. Можно показать, что в сети Петри, построенной из моделей отдельных правил Δ-грамматики, все переходы будут различны. Сформулируем данное утверждение в виде леммы и теоремы.

Лемма 2.1. Пусть Γ^{RL} – расширенная лексико-синтаксическая Δ-грамматика вида (1.2). Все правила $\pi \in \Pi^{R}$ указанной грамматики, относящиеся к произвольным элементарным преобразованиям, являются взаимно различными.

Доказательство леммы следует из доказанной в [7] для синтаксических Δ-грамматик теоремы о моделировании произвольного элементарного преобразования специальными элементарными преобразованиями типа расщепления, перевешивания узла и склеивания узлов дерева.

Лемма 2.2. Проблема достижимости заданной разметки M_i^k из начальной M_{0i} в сети N_i разрешима.

Доказательство леммы следует из конечности дерева достижимости [105,106,117] для ограниченных сетей Петри.

Теорема 2.2. Все символы-переходы $t_i^j \in T_i$ сети N_i различны.

Доказательство следует из доказанной *Леммы 2.1*.

Из доказанной теоремы следует, что помечающая функция $\sum : T_i \to T^A$ для сети N_i сопоставляет каждому переходу $t_i^j \in T_i$ единственный символ алфавита T^A, соответствующий обозначению

некоторого правила из произвольных элементарных преобразований в заданной Δ-грамматике.

Будучи помеченной, сеть N_i обладает рядом свойств, актуальных для задач достижимости заданной разметки и принадлежности произвольной последовательности символов алфавита T^A языку рассматриваемой сети Петри.

Свойство 1. Некоторая фиксированная разметка M_{fi}, называемая терминальной, допустима в сети N_i тогда и только тогда, когда среди множества правил моделируемой системы имеются односторонние преобразования, выходам которых соответствуют тупиковые разметки.

Свойство 2. Свободный терминальный язык $L(N_i, M_{fi})$ сети N_i, описываемый последовательностями переходов от начальной разметки M_{0i} к фиксированной терминальной разметке M_{fi}, определяется в зависимости от задания M_{0i}.

Свойство 3. Произвольность задания начальной разметки M_{0i} влечет возможность существования нескольких свободных терминальных языков $L(N_i, M_{fi})$ на сети N_i.

Свойство 4. Префиксный язык $\{\Sigma(\tau) | \tau \in L(N_i)\}$ помеченной сети (N_i, Σ) получается из свободного языка $L(N_i)$ прямой заменой символов-переходов $t_i^j \in T_i$ на соответствующие символы из T^A.

Свойство 5. В сети N_i, помеченной символами алфавита T^A, появление λ-переходов (переходов, которым не сопоставляется ни один символ из T^A, [39, стр. 36]) возможно возможно в случае моделирования некоторого произвольного элементарного

преобразования последовательностью универсальных специальных элементарных преобразований. В этом случае соответствующий переход $t_i^j \in T_i$ замещается последовательностью λ-переходов, соответствующих выполняемым универсальным элементарным преобразованиям, а префиксный язык $\{\sum(\tau)|\tau \in L(N_i)\}$ помеченной сети $(N_i, \sum)$ будет относиться к классу ℓ^λ префиксных языков сетей Петри. Верхний индекс λ означает, что помечающая функция может быть частичной, то есть помеченная сеть $(N_i, \sum)$ может содержать λ-переходы. При отсутствии λ-переходов ее префиксный язык будет относиться к подклассу ℓ класса ℓ^λ префиксных языков сетей Петри.

Теорема 2.3. Проблема определения обратимости слова $\tau \in T_i^* | M_{0i} \xrightarrow{\tau} M_i^k$ языка $L(N_i)$ разрешима.

Доказательство. При известной последовательности смены разметок и наличии (согласно *Лемме 2.1*) взаимно-однозначного соответствия между указанной последовательностью и последовательностью переходов $\tau' = \left(t_i^{k'}, t_i^{(k-1)'}, \ldots, t_i^{2'}, t_i^{1'}\right)$ задача сводится к определению принадлежности слова τ' языку $L(N_i)$. Префиксный язык $\{\sum(\tau)|\tau \in L(N_i)\}$ помеченной сети $(N_i, \sum)$ относится либо к классу ℓ^λ, либо к подклассу ℓ этого класса языков. Как было доказано в [39, стр. 48], проблема принадлежности разрешима для языков класса ℓ и ℓ^λ, что позволяет говорить о разрешимости задачи поиска обратного слова и для слова $\tau \in T_i^* | M_{0i} \xrightarrow{\tau} M_i^k$.

Теорема 2.4. Проблема определения оптимального слова $\tau \in T_i^* | M_{0i} \xrightarrow{\tau} M_i^k$ (T_i^* - множество всех слов в алфавите T_i) в языке $L(N_i)$ является разрешимой.

Доказательство. Для любой разметки M_i^k в сети N_i проблема ее достижимости из выбранной начальной разметки M_{0i} является разрешимой по *Лемме 2.2*. Определение обратимости слова языка $L(N_i)$ разрешимо по *Теореме 2.3*. Определение минимального среди обратимых слов $\tau \in T_i^* | M_{0i} \xrightarrow{\tau} M_i^k$ делается разрешимым введением функции $f\left(M_i^k\right)$ оценки стоимости пути по дереву достижимости и процедуры отсечения ветвей дерева достижимости, соответствующих найденному решению, во избежание зацикливания алгоритма.

Таким образом, задача приведения Δ-грамматикой входного дерева к целевому виду сводится к задаче достижимости разметки в сети Петри, построенной с использованием моделей правил Δ-грамматики как примитивов, а также исследованию свойств языка такой сети.

С целью сокращения перебора при определении оптимального слова $\tau \in T_i^* | M_{0i} \xrightarrow{\tau} M_i^k$ и учитывая требование равноудаленности целевого состояния от состояний, соответствующих M_{0i} и M_i^k, будем рассматривать состояние системы одновременной активизацией не одного, а двух информационных элементов, соответствующих сценарию.

Действительно, рассмотренная сеть N_i позволяет отражать активизацию только одного информационного элемента, в то время как состояние системы в решаемой задаче построения системы целевых выводов в Δ-грамматике описывается как минимум двумя одновременно активизированными элементами, соответствующими входам-выходам различных правил. Более того, на определенной таким образом информационно-логической модели системы правил Δ-грамматики отсутствует важный компонент, позволяющий задать

реальное целевое состояние системы, которое отличается от описываемого в модели посредством разметки M_i^k в случае, если оптимальное слово $\tau \in T_i^* \left| M_{0i} \xrightarrow{\tau} M_i^k \right.$ является обратимым.

Иными словами, если в системе правил существует минимальная последовательность двусторонних преобразований $\pi_1, \pi_2, \ldots, \pi_{k-1}, \pi_k$, где входам/выходам правил π_1 и π_k функционально соответствуют исходные деревья, то целевое состояние, соответствующее одинаковой ЛСК, должно быть равноудалено и от входа π_1, и от входа π_k. При найденном обратимом оптимальном слове $\tau \in T_i^* \left| M_{0i} \xrightarrow{\tau} M_i^k \right.$ это означает описать последовательности переходов $\tau_1 = \left(t_i^1, t_i^2, \ldots, t_i^l\right)$ и $\tau_1^{'} = \left(t_i^{k'}, t_i^{(k-1)'}, \ldots, t_i^{l'}\right)$, где $\tau_1 \in T_i^* \left| M_{0i} \xrightarrow{\tau_1} M_i^l \right.$, а $\tau_1^{'} \in T_i^* \left| M_i^k \xrightarrow{\tau_1^{'}} M_i^l \right.$.

Последовательности τ_1 и $\tau_1^{'}$ должны удовлетворять следующему требованию: если $|\tau| \bmod 2 = 1$, то $|\tau_1| = |\tau| \; div \; 2 + 1$ и $|\tau_1^{'}| = |\tau| \; div \; 2$, а если $|\tau| \bmod 2 = 0$, то $|\tau_1| = |\tau_1^{'}| = |\tau| \; div \; 2$. Иными словами, на основе найденного обратимого оптимального слова $\tau \in T_i^* \left| M_{0i} \xrightarrow{\tau} M_i^k \right.$, однозначно определяющего достижимость разметки M_i^k из заданной начальной M_{0i} в сети N_i, требуется найти последовательности τ_1 и $\tau_1^{'}$ переходов к разметке M_i^l, равноудаленной от разметок M_{0i} и M_i^k.

Модель системы правил $\pi \in \Pi^R$ переходит из состояния в состояние путем активизации различных пар $\left\{T_1^{\pi}, T_2^{\pi}\right\} \subset T^{\Gamma}$. Сценарии над множеством информационных элементов образуют пары, составляющие подмножество декартова произведения $T^{\Gamma} \times T^{\Gamma}$.

Обозначим множество всех сценариев внутри системы правил, моделируемой сетью N_i, как S_i. Формально пара $\{T_1^\pi, T_2^\pi\}$ соответствует некоторому сценарию $S_i^j \in S_i$. Целевое состояние характеризуется наличием двух фишек в некоторой позиции $p_i^j \in P_i$ и ему соответствует сценарий $S_i^j = \{T_1^\pi, T_2^\pi\}$, для которого $T_1^\pi = T_2^\pi$.

При срабатывании одного из переходов, допустимых в рамках сценария S_i^j, изменяется активность только одного из двух входящих в сценарий информационных элементов. Последовательность переходов от сценария $S_i^j = \{T_l^\pi, T_m^\pi\}$ к сценарию $S_i^k = \{T_n^\pi, T_o^\pi\}$ возможна, если существует пара правил $\{\pi_1, \pi_2\}$ такие, что:

– либо $\pi_1(r_{\ln}): T_l^\pi \xrightarrow{\pi_1(r_{\ln})} T_n^\pi$, $\pi_2(r_{mo}): T_m^\pi \xrightarrow{\pi_2(r_{mo})} T_o^\pi$ и $r_{\ln} \wedge r_{mo} = true$;

– либо $\pi_1(r_{lo}): T_l^\pi \xrightarrow{\pi_1(r_{lo})} T_o^\pi$, $\pi_2(r_{mn}): T_m^\pi \xrightarrow{\pi_2(r_{mn})} T_n^\pi$ и $r_{lo} \wedge r_{mn} = true$.

Теорема 2.5. Для каждого задаваемого над сетью N_i сценария $S_i^j \in S_i$ можно указать максимум два различных перехода $t_i^j \in T_i$ и $t_i^k \in T_i$ такие, что существуют взаимно различные сценарии S_i^{j1} и S_i^{j2} из множества S_i и при этом $S_i^k \xrightarrow{t_i^j} S_i^{j1}$, $S_i^{j1} \xrightarrow{t_i^k} S_i^j$, $S_i^k \xrightarrow{t_i^k} S_i^{j2}$, $S_i^{j2} \xrightarrow{t_i^j} S_i^j$.

Доказательство. Срабатывание одного перехода $t_i^j \in T_i$ ведет к изменению активности максимум одной позиции $p_i^j \in P_i$, поскольку

для $\forall j = 1,\ldots,|T_i| \quad \sum_{k=1}^{|P_i|} H_{jk} = 1$. Согласно *Теореме 2.2*, в сети N_i все символы-переходы различны. Следовательно, два различных сценария над рассматриваемой сетью могут быть связаны только одним переходом. А поскольку число активных позиций сети N_i в рамках сценария равно двум, то существует максимум два пути:

$$S_i^k \xrightarrow{t_i^j} S_i^{j1} \xrightarrow{t_i^k} S_i^j \text{ и } S_i^k \xrightarrow{t_i^k} S_i^{j2} \xrightarrow{t_i^j} S_i^j,$$

где $S_i^{j1} \neq S_i^{j2}$, $t_i^j \neq t_i^k$, $S_i^k \cap S_i^{j1} \neq \varnothing$, $S_i^{j1} \cap S_i^j \neq \varnothing$, $S_i^k \cap S_i^{j2} \neq \varnothing$, $S_i^{j2} \cap S_i^j \neq \varnothing$ (рис. 2.1), что и служит доказательством теоремы.

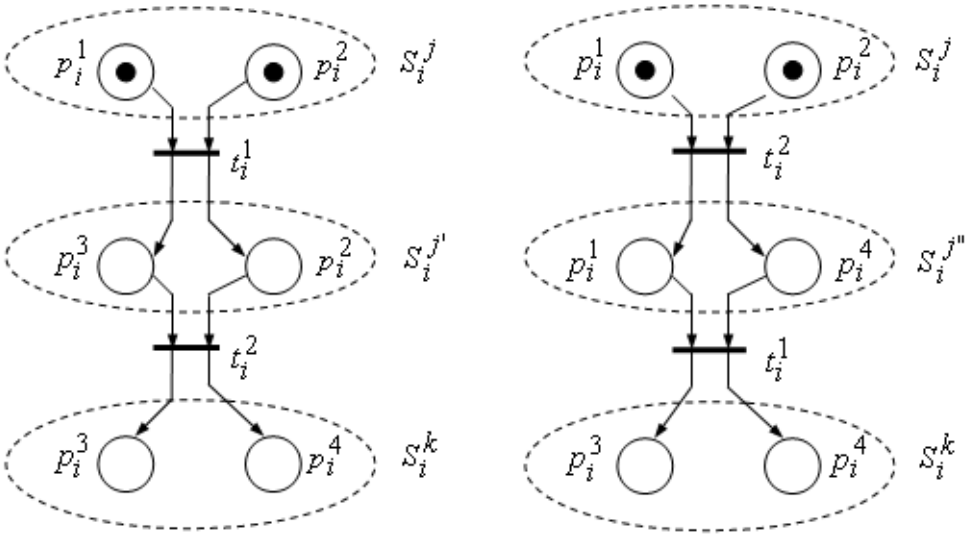


Рис. 2.1. Две возможные последовательности переходов от сценария S_i^j к сценарию S_i^k

При нахождении решения тот информационный элемент, который соответствует целевому состоянию системы, будет активизирован дважды. Поэтому максимальное число сценариев, задаваемых над множеством позиций P_i сети N_i, равно количеству

комбинаций из $|P_i|$ по 2 (случаи активизации различных информационных элементов) плюс мощность множества P_i:

$$\max|S_i| = \frac{|P_i|!}{2(|P_i|-2)!} + |P_i|.$$

Рассмотрим символьное описание сети N_i, допускающее ее машинную обработку и хранение, с использованием представления сценария S_i^j как совокупности двух одновременно активизированных информационных элементов: $S_i^j = \{p_j^1, p_j^2\}$. Поскольку в рамках одного и того же сценария может быть разрешено несколько переходов, то его описание будет выглядеть следующим образом:

$$S_i^j = \{S_i^{j1}, \ldots, S_i^{jk}, p_j^1, p_j^2\}, \qquad (2.12)$$

где $S_i^{j1}, \ldots, S_i^{jk}$ – множество сценариев, связанных с S_i^j;

$\{p_j^1, p_j^2\}$ – множество позиций сети N_i, активных в рамках S_i^j.

Множество сценариев, связанных с S_i^j через разрешенные в рамках сценария переходы $t_i^j \in T_i$, можно представить массивом ссылок $ref_i^j(k_i^j)$. Обозначив $\{p_j^1, p_j^2\}$ как P_i^j, получаем:

$$S_i^j = \{ref_i^j(k_i^j), P_i^j\}. \qquad (2.13)$$

Структура вида $S_i^j = \{ref_i^j, P_i^j\}$ формируется в зависимости от содержимого $P_i^j \subset P_i$ посредством обработки матрицы инцидентности F. Исходными данными при этом будут:

$\Sigma_{Ri} = \{t_i^1, t_i^2, \ldots, t_i^{|T_i|}\}$ – массив информации о переходах. Элементами указанного массива являются ссылки на описания условий применимости правил Δ-грамматики, соответствующих

переходам $t_i^j \in T_i$. Каждое из условий определяется соотношениями (2.6) и (2.7);

$\sum_{dbfi} = P_i = \left\{ p_i^1, p_i^2, \ldots, p_i^{|P_i|} \right\}$ – массив ссылок на описание входов/выходов правил системы.

Описание сценария структурой (2.12) позволяет эффективно организовать поиск сценариев по дереву из заданного S_i^0, но затрудняет обратный просмотр от S_i^j к S_i^0 при формировании пути к целевому состоянию. Кроме того не исключается генерация сценариев $S_i^j \in S_i$ и $S_i^k \in S_i : P_i^j = P_i^k$, лежащих в дереве на одном пути от S_i^0. В силу сказанного, а также принимая во внимание вытекающую из *Теоремы 2.5* единственность последовательности смены сценариев от S_i^0 к S_i^j, введем альтернативное описание сценария $S_i^j \in S_i$:

$$S_i^j = \left\{ ref\_back_i^j, P_i^j \right\}, \qquad (2.14)$$

где P_i^j определяется аналогично соответствующему компоненту структуры (2.13), $ref\_back_i^j$ – ссылка на сценарий, с которым S_i^j связан посредством некоторого перехода $t_i^j \in T_i$, инцидентного одной из позиций, входящих в P_i^j.

При описании сценария в виде (2.14) поиск целевого сценария, удовлетворяющего условию $P_i^j = \left\{ p_j^1, p_j^2 \right\} : p_j^1 = p_j^2$, на основе заданного начального $S_i^0 = \left\{ nil, \left\{ p_0^1, p_0^2 \right\} \right\}$ организуется как генерация списка $ref_i^j\left(k_i^j\right)$ каждого сценария S_i^j очередного уровня дерева сценариев с попутной проверкой условия $p_j^1 = p_j^2$ для каждого создаваемого сценария. В случае отсутствия решения на созданном

уровне дерева сценариев процедура повторяется для полученных списков. Для каждого найденного решения запоминается путь по дереву от S_i^0.

Задавая сеть N_i парой массивов $\Sigma = (\Sigma_{Ri}, \Sigma_{dbfi})$, можно описать динамику функционирования системы правил Δ-грамматики построением *TS*-сети (ограниченной сети Петри, порождаемой множеством символов-переходов T_i на множестве сценариев S_i [106]) на основе задаваемой начальной разметки. При этом указанная разметка соответствует активизации пары позиций сети N_i для входов правил, которым функционально соответствуют исходные деревья T_1^{π} и T_2^{π}. В следующем разделе мы рассмотрим взаимосвязь внутренней структуры входов/выходов правил Δ-грамматики как объектов информационного пространства с информационным наполнением деревьев глубинного синтаксиса.

2.3. Моделирование построения образа суммарного смысла

Предложенная в предыдущем разделе модель учитывает недетерминированный характер порождения Δ-грамматикой множества деревьев, при этом построение целевого вывода сводится к классическим задачам теории сетей Петри. Однако рассмотрение входа/выхода правила в качестве объекта информационного пространства требует формального описания его активизации в зависимости от ситуации использования и с учетом его внутренней структуры. Сказанное предполагает решение двух основных задач:

– построение модели входа/выхода правила как объекта информационного пространства;

– разработка структуры информационного наполнения анализируемого дерева.

При этом основным требованием к модели входа/выхода правила является отображение различных способов его использования при единообразии функционального описания. Анализ вызывающих активизацию входа/выхода правила событий позволяет выделить следующие способы его использования как информационного элемента:

– анализ применимости правила к помеченному дереву с выдачей FALSE/TRUE в качестве результата;
– синтез дерева по задаваемому выходным деревом шаблону;
– распознавание ключевого слова заменяемого лексическим правилом поддерева;
– расстановка композиционных меток в анализируемом дереве с целью обозначения заменяемого поддерева.

Во всех четырех показанных ситуациях элементы информационного пространства активизируются по-разному ввиду неоднородности вызывающих их активизацию событий при идентичности функциональной структуры процессов активизации. Поскольку задача применения правила к некоторому заданному дереву есть частный случай задачи “Изоморфизм подграфу” [112], то логико-функциональная структура информационного наполнения входного/выходного дерева правила должна быть идентична логико-функциональной структуре информационного наполнения анализируемых деревьев. Говоря об изоморфизме поддереву, будем подразумевать изоморфизм с точностью до функционального соответствия. Само функциональное соответствие определим следующим образом.

Определение 2.7. Деревья T_1 и T_1' считаются изоморфными с точностью до функционального соответствия, если в дереве T_1' из узла α_{11}' в узел α_{12}' идет ветвь с некоторой пометкой тогда и только тогда, когда в дереве T_1 из узла α_{11} в узел α_{12} идет ветвь с той же пометкой. При этом узел α_{11}' должен отвечать требованиям, содержащимся в узле α_{11}, а узел α_{12}', соответственно, требованиям, содержащимся в узле α_{12}. В таком случае считается, что узел α_{11}' функционально соответствует узлу α_{11}, а узел α_{12}' – узлу α_{12}.

Рассмотрим структуру информационного наполнения узла дерева на входе/выходе правила, унифицируемую со структурой соответствующего описания для анализируемых деревьев и ориентированную на представление динамических структур данных средствами декларативных языков.

В соответствии с приведенным в работах И.А.Мельчука описанием уровня глубинного синтаксиса, в информационном наполнении узла глубинной синтаксической структуры следует выделить:

- лексическую часть, соответствующую представленному в узле элементу множества W^R модели (1.2);
- грамматическую часть, содержащую семантические словоизменительные характеристики.

Кроме того, в описание узла должны быть введены особые элементы, соответствующие пометке входящей в узел ветви и композиционной метке.

Представим дерево глубинного синтаксиса фразы χ упорядоченной двойкой

$$T_\chi = \langle W_\chi, V_\chi \rangle, \qquad (2.15)$$

где W_χ есть множество узлов, а V_χ есть множество ветвей дерева. Информационное наполнение отдельного узла $w_{\chi i} \in W_\chi$ может быть представлено списком из четырех элементов:

$$w_{\chi i} = \left(lx_{\chi i}, gr_{xi}, ar_{\chi i}, cl_{\chi i}\right). \qquad (2.16)$$

Здесь элемент $lx_{\chi i}$ соответствует лексической, $gr_{\chi i}$ – грамматической части узла, $ar_{\chi i}$ – пометке входящей ветви, а $cl_{\chi i}$ – композиционной метке узла. Следует отметить, что $cl_{\chi i}$ является необязательным (факультативным) элементом в структуре (2.16) и вводится для обозначения того факта, что рассматриваемый узел является выделенным и участвует в некотором преобразовании исходного дерева.

Как показано в [15], дерево T_2 получается из дерева T_1 применением элементарного преобразования $t_1 \Rightarrow t_2 \mid f$ при задаваемом функцией f однозначном отображении множества узлов дерева t_1 во множество узлов дерева t_2, если T_1 и T_2 представимы, соответственно, в виде:

$$T_1 = C\left(T^0; \alpha_0 \middle| C\left(t_1; \alpha_1, \alpha_2, \ldots, \alpha_n \middle| T^1, T^2, \ldots, T^n\right)\right), \qquad (2.17)$$

$$T_2 = C\left(T^0; \alpha_0 \middle| C\left(t_2; f(\alpha_1), f(\alpha_2), \ldots, f(\alpha_n) \middle| T^1, T^2, \ldots, T^n\right)\right), \qquad (2.18)$$

C – операция композиции. Она определяется следующим образом. Пусть в дереве T^0 выделен узел α_0, а в дереве t_1 выделено n узлов $\alpha_1, \alpha_2, \ldots, \alpha_n$ (не обязательно попарно различных). Тогда дерево T_1 получается из T^0 в два этапа: “наклеивание” вершин деревьев $T^1, T^2, \ldots, T^n$ на узлы $\alpha_1, \alpha_2, \ldots, \alpha_n$ дерева t_1 и последующее

“наклеивание” вершины получившегося дерева на узел α_0 дерева T^0. Будем в дальнейшем называть дерево T^0 деревом верхнего контекста (верхним древесным контекстом) заменяемого правилом дерева t_1, а деревья $T^1, T^2, \ldots, T^n$ - деревьями, соответственно, нижнего контекста (нижним древесным контекстом) заменяемого дерева.

В том случае, когда узел $w_{\chi i} \in W_\chi$ является выделенным, композиционная метка $cl_{\chi i}$ присутствует в структуре (2.16) и принимает значения:

- равное 0 для дерева T^0 и обозначает место “крепления” заменяемого (t_1) и заменяющего (t_2) деревьев к T^0;
- в диапазоне от 1 до n – для деревьев нижнего контекста. Каждая из меток $1, \ldots, n$ обозначает место крепления соответствующего дерева $T^1, T^2, \ldots, T^n$ к заменяемому (заменяющему) дереву.

Лексическая часть $lx_{\chi i}$ узла $w_{\chi i} \in W_\chi$ представляется списочной структурой вида:

$$lx_{\chi i} = (C_0, fun_n, \ldots, fun_1),$$

где C_0 представляет некоторую самостоятельную лексему, лексической производной от которой (в виде последовательно взятых значений лексических функций из списка $fun_n, \ldots, fun_1$) является лексема, соответствующая содержимому узла на поверхностно-синтаксическом уровне. При этом список $fun_n, \ldots, fun_1$ может быть пустым в случае отображения в узле фиктивной лексемы, идиомы, либо самостоятельной лексемы, не являющейся лексическим

коррелятом других лексем, присутствующих в той же глубинной синтаксической структуре.

Грамматическая часть $gr_{\chi i}$ узла $w_{\chi i} \in W_{\chi}$ представляется упорядоченной двойкой:

$$gr_{\chi i} = (psp, lstsc),$$

где psp – символьное обозначение части речи (табл. 2.1), $lstsc$ – список семантически обусловленных словоизменительных категорий, обсуждавшихся в [45, стр. 144]. У существительных к числу таковых относится число, у глаголов – вид, время, наклонение.

Таблица 2.1

Символьные обозначения частей речи

psp	часть речи	*psp*	часть речи
S	существительное	Conj	союз
V	глагол	Num	числительное
A	прилагательное	P	причастие
Adv	наречие	Prep	предлог

Элемент $ar_{\chi i}$ в составе структуры (2.16) принимает целочисленные значения одного из шести типов связей между родительским и дочерним узлом в глубинной синтаксической структуре, а для вершины дерева $ar_{\chi i}$ имеет значение 0 (входящая ветвь отсутствует).

Описание информации узла $w_{\chi i} \in W_{\chi}$ в виде списка (2.16) позволяет:

– формально определить функциональные требования к узлу ГСС при описании компонент дерева, заменяемого

некоторым лексическим правилом. При этом символ C_0 выступает в качестве служебного: им задается местонахождение ключевого слова ЛСК;

– при реализации рассматриваемых преобразований деревьев на языке Лисп организовать вычисление значения суперпозиции лексических функций из списка $fun_n,\ldots, fun_1$ с использованием их имен в качестве функциональных аргументов;

Если дерево глубинного синтаксиса фразы χ представить упорядоченной двойкой вида (2.15), то для машинного представления входа/выхода некоторого правила исследуемой Δ-грамматики в целях учета динамики процесса применения этого правила к конкретному дереву целесообразно ввести структуру следующего вида:

$$T_k^{\pi} = \left\langle W_k^{\pi}, V_k^{\pi}, A_k^{\pi} \right\rangle, \qquad (2.19)$$

где W_k^{π} есть множество требований к содержимому узлов, V_k^{π} – множество требований к разметке ветвей дерева. Компонент A_k^{π} в терминологии теории графов есть матрица смежности, каждый элемент A_{kij}^{π} принимает одно из двух возможных значений:

– 1, если в дереве существует ветвь от узла w_i^{π} к узлу w_j^{π}, где $\left\{w_i^{\pi}, w_j^{\pi}\right\} \subset W_k^{\pi}$;

– 0 – в противном случае.

Само дерево при этом может быть представлено рекурсивной структурой данных, каждый элемент которой будет содержать описание вершины в виде (2.16) и список дочерних поддеревьев.

В качестве примера на рис. 2.2 приведено списочное описание (в нотации Microsoft muLISP) для входа лексического правила №17 с обслуживающим его синтаксическим правилом №6 в составе системы синонимического перифразирования русского языка[1].

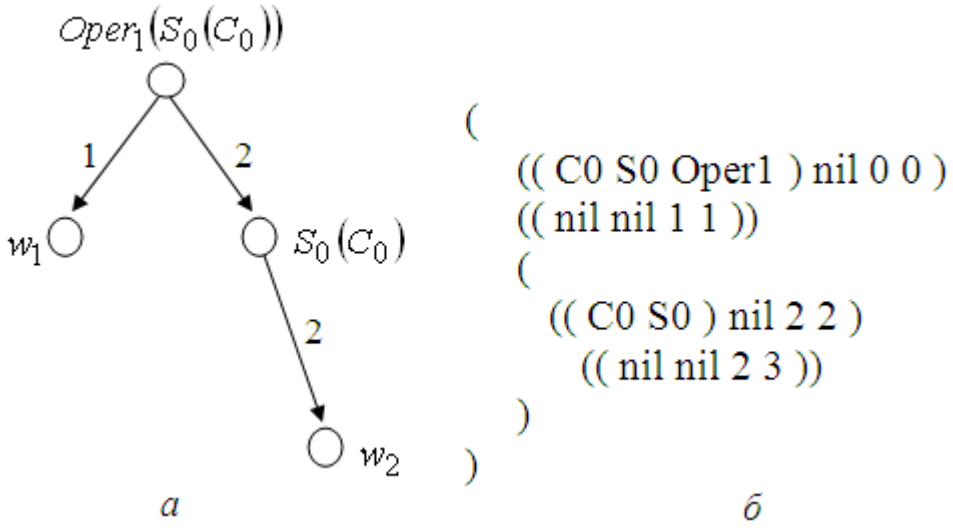


Рис. 2.2. Входное дерево правила Δ-грамматики:

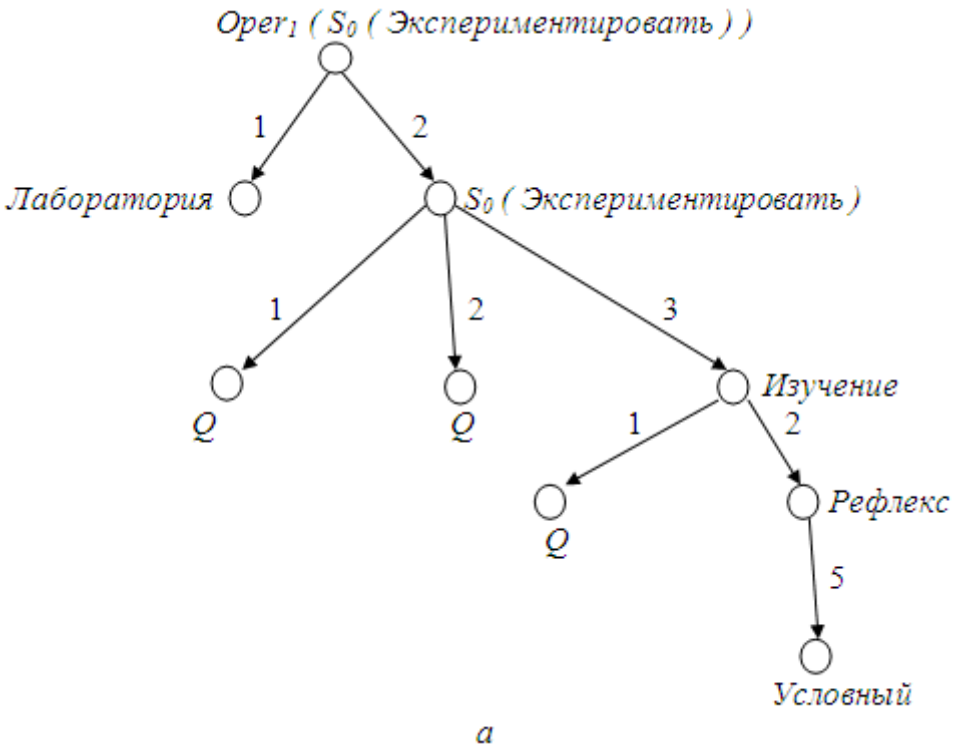
а – графическое представление[2]; б – списочное описание в нотации языка Лисп

Как видно из указанного примера, существенной особенностью представления информации узлов входного дерева правила Δ-грамматики является отсутствие определения отдельных компонент дерева. В частности, это относится к требованиям, предъявляемым к лексической и грамматической части узлов входного дерева синтаксическими правилами, обслуживающими лексические замены. В таком случае считается, что соответствующий компонент структуры вида (2.16) имеет пустое или неопределенное значение, то есть *nil*.

[1] См. [17, стр. 154].

[2] Узлы w_1 и w_2 соответствуют произвольным словам, не меняющимся в процессе синонимического перифразирования.

Действительно, в общем случае лексическое синонимическое преобразование дерева глубинного синтаксиса обслуживается одним или несколькими синтаксическими преобразованиями. Поэтому входное дерево для лексического преобразования следует рассматривать как поддерево входного дерева первого из обслуживающих данную лексическую замену синтаксических преобразований. При этом для синтаксических преобразований значимой является только разметка ветвей, чем и обусловлено присутствие *nil* в качестве значения лексической и грамматической части описания узлов, не входящих в ЛСК. Для сравнения на рис. 2.3 приведено дерево глубинной синтаксической структуры простого распространенного предложения русского языка “*Лаборатория провела эксперименты по изучению условных рефлексов*”.



а

Рис. 2.3. Анализируемое дерево глубинного синтаксиса:

а – графическое представление; б – списочное описание в нотации языка Лисп

```
(
 (( Экспериментировать S0  Oper1 )
    (V (сов_вид прош_вр изъявит_накл )) 0 nil
 )
   ((( Лаборатория nil )( S (ед_ч)) 1 nil ))
   ((( Экспериментировать S0 )( S (мн_ч)) 2 nil )
    ((( Q nil)( S nil ) 1 nil ))
    ((( Q nil )( S (на)) 2 nil ))
    ((( Изучение nil )( S (по ед_ч)) 3 nil)
      (((Q nil) nil 1 nil))
      ((( Рефлекс nil)( S (мн_ч)) 2 nil)
        (( Условный nil )(A (мн_ч)) 5 nil)
      )
    )
   )
)
```

б

Рис. 2.3. Продолжение

Поскольку анализ применимости правил Δ-грамматики для данного предложения не проводился, композиционные метки не определены (в соответствующих местах описания структуры (2.16) на рис. 2.3, *б* стоит *nil*).

При наличии для структур (2.19) и (2.15) списочного описания, представленного на рис. 2.2, *б* и рис. 2.3, *б*, структура вида (2.19) может рассматриваться как система, порождающая отличные друг от друга процессы с идентичной функциональной структурой. Прохождение отдельного узла $w_i^{\pi} \in W_k^{\pi}$ при рекурсивной обработке может быть рассмотрено как абстрактное событие, а установление функционального соответствия некоторого узла анализируемого дерева требованиям узла w_i^{π}, размещение в анализируемом узле композиционной метки, синтез дерева по представляемому

посредством w_i^{π} шаблону – как разные варианты реализации этого события. Единообразие функционального описания входа/выхода правила позволяет рассматривать и анализ применимости правила π, и синтез дерева, соответствующего выходу правила, как процессы, порождаемые одной и той же сетью Петри, описывающей функциональную схему представленного в виде дерева (2.19) входа/выхода правила π:

$$N_{\pi(k)} = \left\{P_{\pi(k)}, T_{\pi(k)}, F_{\pi(k)}, H_{\pi(k)}, C, M_{0\pi(k)}\right\}, \qquad (2.20)$$

где множество позиций $P_{\pi(k)}$ соответствует множеству состояний информационного элемента, а каждое состояние отождествляется с очередным пройденным узлом $w_i^{\pi} \in W_k^{\pi}$; каждому из переходов $t_{\pi(k)}^{i} \in T_{\pi(k)}$ соответствует совокупность требований лексической, грамматической части и метки входящей ветви узла w_i^{π}; $F_{\pi(k)}$ и $H_{\pi(k)}$ есть матрицы инцидентности, аналогичные соответствующим компонентам структуры (2.8); $C = \{c_1, c_2, c_3, c_4, c_5\}$ – множество цветов маркера; $M_{0\pi(k)}$ – начальная разметка.

Каждому из цветов маркера соответствует определенный способ использования информационного элемента как вариант разовых реализаций событий прохождения узлов $w_i^{\pi} \in W_k^{\pi}$ при обходе дерева T_k^{π}, а именно: c_1 – анализ применимости правила, c_2 – синтез дерева на выходе правила, c_3 – определение ключевого слова ЛСК, c_4 – расстановка композиционных меток в анализируемом дереве T_{χ}.

Следует отметить важные особенности сети (2.20), актуальные для моделирования активизации дерева T_k^{π} как объекта информационного пространства с учетом последовательности действий в процессах, порождаемых входом/выходом правила π.

Каждый пройденный узел $w_i^{\pi} \in W_k^{\pi}$ представляется как разовая реализация факта изменения некоторого условия в системе (в сети $N_{\pi(k)}$ указанным изменениям соответствуют элементы множества $P_{\pi(k)}$), а анализ требований лексической, грамматической части и метки входящей ветви узла w_i^{π} – как действие в процессе прохождения дерева T_k^{π}.

Для обозначения действия, связанного с окончанием обхода дерева T_k^{π}, множество $T_{\pi(k)}$ содержит особый переход $t_{out} \Leftrightarrow t_{\pi(k)}^{\left|T_{\pi(k)}\right|}$, инцидентный всем позициям $p_{\pi(k)}^{i} \in P_{\pi(k)}$: $\sum_{j=1}^{\left|T_{\pi(k)}\right|} F_{\pi(k)ij} = 0$. Аналогично для обозначения изменения соответствующего условия в множество позиций сети (2.20) введена позиция $p_{out} \Leftrightarrow p_{\pi(k)}^{\left|T_{\pi(k)}\right|}$, инцидентная единственному переходу t_{out}.

В случае успешного анализа применимости правила π к дереву T_{χ} последующая перестройка последнего требует идентификации ключевого слова заменяемой ЛСК и расстановки композиционных меток. Для задания последовательности указанных процессов в структуру сети (2.20) введена дополнительная дуга, соединяющая переход t_{out} с позицией $p_{\pi(k)}^{1}$, соответствующей началу обхода

дерева T_k^{π} на входе/выходе правила. С целью формализации условия окончания анализа/синтеза (во избежание развертывания бесконечных процессов в сети) множество C содержит нейтральный маркер c_5, запрещающий срабатывание перехода, а для перехода t_{out} задается индивидуальная таблица условий срабатывания (табл. 2.2).

Таблица 2.2

Условия срабатывания перехода t_{out}

$p^i_{\pi(k)} \in P_{\pi(k)} : F_{\pi(k)ij} = 1$ для $j = \left\|T_{\pi(k)}\right\|$	p_{out}	$p^1_{\pi(k)}$
c_1	c_3	c_3
c_3	c_4	c_4
c_4	c_5	c_5
c_2	c_5	c_5

Для разрешения конфликтных ситуаций (когда реализация одного события системы исключает возможность реализации других, [70, стр. 44]) при сетевом моделировании рекурсивной обработки леса дочерних поддеревьев узла $w_i^{\pi} \in W_k^{\pi}$ топология исходной сети вида (2.20) преобразуется путем замены участка сети, включающего позицию $p^i_{\pi(k)}$ и инцидентные ей конфликтующие переходы $t^j_{\pi(k)}$ и $t^l_{\pi(k)}$ по правилу, показанному на рис. 2.4. Здесь добавляемый переход $t^{i'}_{\pi(k)}$ есть безусловный переход, инцидентный позициям $p^{i'}_{\pi(k)}$ и $p^{i''}_{\pi(k)}$, каждая из которых представляет собой копию позиции $p^i_{\pi(k)}$.

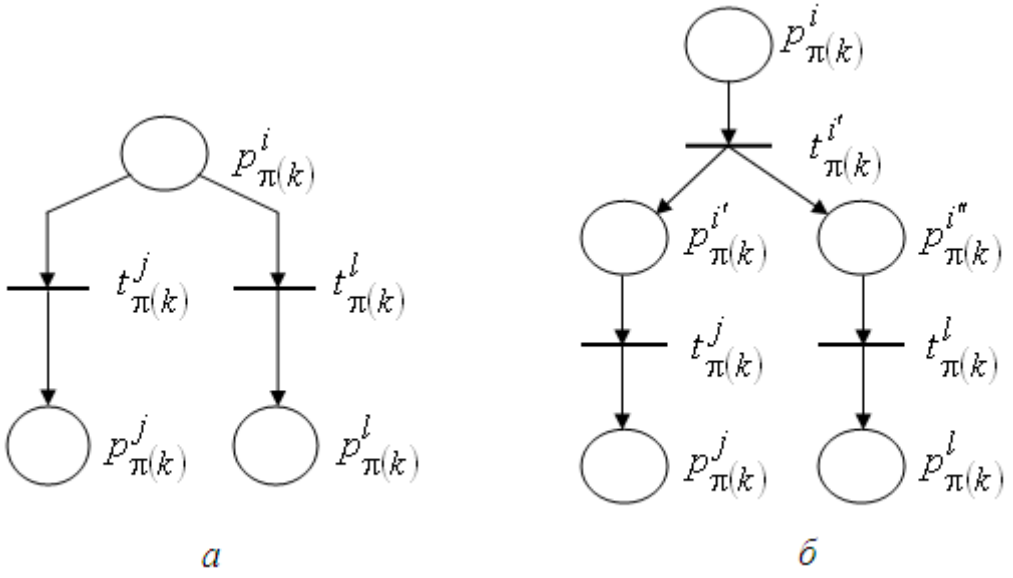


Рис. 2.4. Разрешение конфликта в сети вида (2.20) преобразованием топологии:

а – фрагмент сети до преобразования; б – преобразованный фрагмент

Пример сетевой модели для представленного на рис. 2.2, *а* входа правила показан на рис. 2.5.

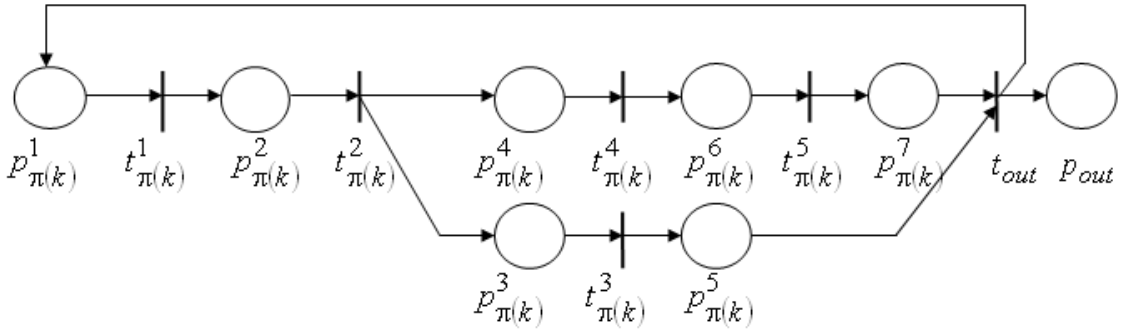


Рис. 2.5. Сетевая модель входа/выхода правила:

переход $t^{1}_{\pi(k)}$ соответствует прохождению вершины, $t^{3}_{\pi(k)}$ – узла w_1, $t^{4}_{\pi(k)}$ – узла с содержимым $S_0(C_0)$, $t^{5}_{\pi(k)}$ – узла w_2

Сеть $N_{\pi(k)}$ обладает рядом свойств, позволяющих оценить адекватность порождаемых ей процессов моделируемым процессам, порождаемым входом/выходом правила π заданной Δ-грамматики как

системой при анализе применимости правила к некоторому дереву, либо синтезе результирующего дерева по шаблону, определяемому посредством T_k^{π}.

Теорема 2.6. Все порождаемые сетью $N_{\pi(k)}$ процессы конечны.

Доказательство следует из конечности (по определению) множеств позиций и переходов сети, а также наложенных *Таблицей 2.2* ограничений на срабатывание перехода t_{out}.

Теорема 2.7. Сеть $N_{\pi(k)}$ является ограниченной.

Доказательство. Как следует из *Теоремы 2.6*, любая позиция $p_{\pi(k)}^{j} \in P_{\pi(k)}$ может содержать максимум по одному маркеру каждого из цветов $\overset{5}{\underset{i=1}{c_i}} \in C$. При этом максимальное количество маркеров в позиции равно трем (для позиции p_{out}), что и служит доказательством ограниченности сети $N_{\pi(k)}$.

Таким образом, сетью $N_{\pi(k)}$ порождаются конечные параллельные процессы без альтернатив и конкуренции. Появление в позиции p_{out} одновременно маркеров цветов c_3, c_4 и c_5 (при анализе применимости правила π), либо одновременно маркеров цветов c_2 и c_5 (при синтезе дерева по шаблону, задаваемому деревом T_k^{π}) соответствует завершению указанных процессов. При этом активизация самого T_k^{π} как объекта информационного пространства может быть формально определена как достижение тупиковой разметки в сети $N_{\pi(k)}$ при успешном завершении процесса анализа/синтеза.

Представление анализа входного дерева, либо синтеза дерева, получаемого на выходе правила, как процесса, порождаемого сетью Петри, позволяет:

- фиксировать историю процесса анализа применимости правила к дереву расстановкой композиционных меток в узлах для последующего развертывания синтеза дерева, соответствующего выходу правила;
- унифицировать математический аппарат, применяемый для анализа и синтеза дерева в рамках одного и того же сетевого формализма.

Для анализа смысловой взаимной дополняемости глубинных синтаксических структур $T_{\chi 1}$ и $T_{\chi 2}$ фраз $\chi 1$ и $\chi 2$ в соответствии с *Определением 2.5* после анализа применимости правил некоторой заданной Δ-грамматики с построением последовательности преобразований ЛСК требуется сравнить результаты декомпозиции обоих деревьев. При этом согласно соотношениям (2.17) и (2.18) выполняется сравнение следующих поддеревьев:

- деревьев, замененных совместной работой лексических правил и обслуживающих их синтаксических замен ($t_1^{\chi 1}$ и $t_1^{\chi 2}$);
- деревьев верхнего контекста для заменяемых правилами деревьев $t_1^{\chi 1}$ и $t_1^{\chi 2}$ (соответственно, $T_{\chi 1}^0$ и $T_{\chi 2}^0$);
- множеств деревьев нижнего контекста деревьев $t_1^{\chi 1}$ и $t_1^{\chi 2}$ ($T_{\chi 1}^1, T_{\chi 1}^2, \ldots, T_{\chi 1}^n$ и $T_{\chi 2}^1, T_{\chi 2}^2, \ldots, T_{\chi 2}^n$, соответственно).

На основе *Определения 2.5* введем понятие функционального соответствия для узлов суммируемых ГСС, представляемых структурами вида (2.15).

Определение 2.8. Будем считать, что узел $w^1_{\chi i} \in W_{\chi 1}$ ГСС $T_{\chi 1} = \langle W_{\chi 1}, V_{\chi 1} \rangle$ функционально соответствует узлу $w^2_{\chi i} \in W_{\chi 2}$ ГСС $T_{\chi 2} = \langle W_{\chi 2}, V_{\chi 2} \rangle$, если при описании информационного наполнения этих узлов структурами вида (2.16) не будут выполняться следующие условия:

– $\left(gr^1_{xi} \neq gr^2_{xi}\right) \vee \left(ar^1_{xi} \neq ar^2_{xi}\right) = true$;

– $\left(lx^1_{xi} \neq lx^2_{xi}\right) \wedge \left(lx^1_{xi} \neq Q\right) \wedge \left(lx^2_{xi} \neq Q\right) = true$.

Теорема 2.8. Задача установления функционального соответствия деревьев $T_{\chi 1}$ и $T_{\chi 2}$ принадлежит классу P комбинаторных задач с временной оценкой n^D, где $n = \max\left(\left|W_{\chi 1}\right|, \left|W_{\chi 2}\right|\right)$, $D = \sum_{i=1}^{\left|V^R\right|} \varphi(a_i)$, φ есть матрица вида (1.3), задающая ограничения на характер ветвления в дереве глубинного синтаксиса, V^R – словарь пометок на ветвях.

Доказательство теоремы производится через сведение рассматриваемой задачи к известной NP-полной задаче "Изоморфизм подграфу" [10, стр. 252].

Заметим, что, как следует из *Определения 2.5*, семантическая взаимная дополняемость ЕЯ-фраз на уровне глубинного синтаксиса является относительной. Фактически это означает, что к одной и той же ГСС могут быть применены несколько различных правил преобразования и относительно разных ЛСК. Причем часть из

трансформированных и приведенных к виду с единой ЛСК пар глубинных синтаксических структур не подлежит суммированию ввиду функционального несоответствия друг другу согласно *Определению 2.8*. Более того, среди ряда допустимых вариантов требуется выбрать пару ГСС, для которой достигается максимум “заполнения мест” в соответствии с *Определением 2.6*. Показанная относительность семантической взаимной дополняемости требует рассмотрения функционирования предложенной и исследованной логической модели системы правил Δ-грамматики в плане:

- активизации взаимно различных информационных элементов применительно к одной и той же ГСС;
- формированием множеств ГСС, ЛФ-синонимичных каждой из суммируемых ГСС при приведении последних к виду с одинаковой ЛСК.

Использование сгенерированных таким образом ЛФ-синонимических множеств в задаче установления семантической эквивалентности сравниваемых текстов как основной задаче позволяет уйти от неизбежного увеличения затрат памяти ЭВМ и машинного времени для решения основной задачи при использовании предлагаемого метода распознавания семантических повторов. Эти вопросы освещаются в следующем разделе.

2.4. Служебная информация правил и относительность синонимических преобразований деревьев глубинного синтаксиса

Как было показано в разделе 2.1, к одному и тому же дереву глубинного синтаксиса может быть применено несколько правил синонимических замен. В рамках предложенной информационно-

логической модели сказанное означает активизацию различных элементов информационного пространства применительно к одной и той же ГСС. Описанная в разделе 2.3 функционально-логическая модель входа/выхода правила Δ-грамматики адекватно отображает различные ситуации его использования как информационного элемента, но не учитывает преобразования, примененные к дереву ранее. В содержательной лингвистической интерпретации это означает невозможность применения правила ко второму и последующему вхождениям заменяемого правилом поддерева в анализируемую ГСС. Сказанное особенно актуально при использовании одних и тех же преобразований как для распознавания сверхфразовых единств в анализируемом тексте, так и при установлении его семантической эквивалентности некоторому другому тексту. В настоящем разделе делается попытка уйти от указанного недостатка предложенной модели путем детализации информации, заносимой при работе правил в анализируемые деревья глубинного синтаксиса.

Действительно, результатом анализа применимости некоторого правила к дереву T_χ будет заполнение полей $cl_{\chi i}$ структуры (2.16) для узлов, выделяемых этим преобразованием. Учитывая возможность применения нескольких правил $\{\pi_j,\ldots,\pi_k\}\subset\Pi^R$ синонимических замен к одному и тому же дереву T_χ, при задании композиционной метки $cl_{\chi i}$ узла следует указывать правило, выделяющее этот узел:

$$cl_{\chi i}=\left(\left(cl_{\chi ij},\pi_j\right),\quad\ldots\quad,\left(cl_{\chi ik},\pi_k\right)\right), \tag{2.21}$$

а с учетом возможности применения правила к различным частям одного и того же дерева

$$cl_{\chi i}=\left(\left(cl_{\chi ij},cnt(j),\pi_j\right),\quad\ldots\quad,\left(cl_{\chi ik},cnt(k),\pi_k\right)\right), \tag{2.22}$$

где $cnt(j)$ и $cnt(k)$ представляют собой значения счетчика вхождений в дерево T_χ поддеревьев, изоморфных тем поддеревьям, которые заменяются правилами π_j и π_k, соответственно. При этом изоморфизм устанавливается с точностью до функционального соответствия согласно *Определению 2.7*.

Аналогично списку (2.21) преобразуется список (2.1):

$$\left\{(\pi_i, cnt(i), C_0(i, cnt(i))) : i = 1, \ldots, \left|\Pi^R\right|\right\}, \qquad (2.23)$$

где C_0 есть ключевое слово соответствующей ЛСК.

Список (2.23) формируется в процессе работы сети (2.20) при цветах маркера c_1 и c_3, а элементы списка (2.21) – в ходе следующего прохода той же сети при цвете маркера c_4.

Использование списка (2.22) при анализе применимости правила с расстановкой композиционных меток позволяет избежать зацикливания процесса анализа на одном правиле Δ-грамматики. Действительно, если при цвете маркера c_4 с каждым переходом сети (2.20) связать проверку наличия для узла $w_{\chi i} \in W_\chi$ элементов $(cl_{\chi ij}, cnt(j), \pi_j) \in cl_{\chi i}$, для которых $cl_{\chi ij}$ совпадает с добавляемой композиционной меткой, то повторное выделение в анализируемом дереве T_χ одного и того же поддерева, заменяемого одним и тем же правилом, будет невозможно – процесс остановится на вершине заменяемого поддерева.

Формирование списка вида (2.22) для каждого из узлов, выделяемых в дереве T_χ, согласуется с формированием списка (2.23) следующим образом.

Элемент списка (2.23), относящийся к некоторому правилу, формируется в случае успешного завершения анализа применимости этого правила и занесения в поле $cl_{\chi i}$ структуры (2.16) каждого из выделенных узлов заменяемого правилом поддерева информации в списочной форме (2.22), чему соответствует появлению в позиции p_{out} сети (2.20) одновременно маркеров цветов c_3, c_4 и c_5.

Если содержать в списке (2.23) информацию только о тех правилах, которые не были применены ранее к дереву, то на случай ложной взаимной дополняемости деревьев $T_{\chi 1}$ и $T_{\chi 2}$ исключается повторный поиск правил, применимых к указанным деревьям при построении оставшейся части ЛФ-синонимических множеств для $T_{\chi 1}$ и $T_{\chi 2}$.

Выделяя заменяемые поддеревья по композиционным меткам вида (2.22), можно последовательно относительно разных пар ЛСК определять наличие взаимной дополняемости $T_{\chi 1}$ и $T_{\chi 2}$ на случай ее отсутствия относительно первой из рассматриваемых пар ЛСК. Тем не менее, для корректного взаимодействия процессов увеличения полноты смыслового описания и установления семантической эквивалентности текстов нужно учитывать качественный состав ЛФ-синонимических множеств с точки зрения типов синонимических преобразований, выполняемых при их построении.

Рассмотрим типы преобразований деревьев, допускаемых Δ-грамматикой (1.2) с точки зрения построения целевых выводов, отвечающих требованию обратимости.

Процедура Q_U в составе концептуальной модели (2.3) будет способна строить обратимые выводы, если каждое из используемых ею правил:

– выполняется в обе стороны;

– не ведет к утрате реально выраженных актантов.

Из представленных в [45, стр. 152-159] перечня лексических правил первому требованию не отвечают смысловые импликации (правила № 49-56). Лексические правила № 7,8 и 9 выполняются в обе стороны, однако их применение процедурой Q_U исключено ввиду того, что описываемые ими конверсивные замены ведут к утрате места (валентности) в перерабатываемой ГСС. Корректное применение указанных правил возможно лишь тогда, когда отпадающая валентность в перерабатываемой ГСС не была заполнена.

Пусть $\Pi_{LSC}^{R} \subset \Pi^{R}$ есть множество правил Δ-грамматики (1.2), удовлетворяющих вышеуказанным требованиям.

Теорема 2.9. Построение обратимых выводов процедурой Q_U возможно только с применением правил из множества Π_{LSC}^{R}.

Доказательство теоремы естественным образом вытекает из рассмотренных в *Разделе 2.2* свойств языка сети, моделирующей систему правил Δ-грамматики. При ограничении Δ-грамматикой (1.2) рассмотрением правил множества Π_{LSC}^{R} любое слово в языке указанной сети будет обратимым.

Таким образом, при выделении сверхфразовых единств на множестве деревьев глубинного синтаксиса в соответствии с *Определением 2.6* следует использовать правила множества Π_{LSC}^{R}.

Обозначим множества, порождаемые Δ-грамматикой (1.2) для деревьев $T_{\chi 1}$ и $T_{\chi 2}$ применением правил из Π_{LSC}^{R} относительно некоторого фиксированного ключевого слова, как $T_{\chi 1}^{LSC}$ и $T_{\chi 2}^{LSC}$, соответственно. Тогда в случае отсутствия пары деревьев

$T_{\chi 1i}^{LSC} \in T_{\chi 1}^{LSC}$ и $T_{\chi 2\,j}^{LSC} \in T_{\chi 2}^{LSC}$, для которых возможно построение формального образа сверхфразового единства в соответствии с *Определением 2.6*, впоследствии, уже в процессе установления эквивалентности каждой из фраз $\chi 1$ и $\chi 2$ заданному эталону, в множества $T_{\chi 1}^{LSC}$ и $T_{\chi 2}^{LSC}$ будут заноситься деревья, получаемые из $T_{\chi 1}$ и $T_{\chi 2}$ применением правил $\pi \in \Pi_{LSC}^{R}$, упоминаемых в списках вида (2.23) для $T_{\chi 1}$ и $T_{\chi 2}$, соответственно, и не использованных при приведении этих деревьев к виду с одинаковой ЛСК. А поскольку перестройке подлежит только заменяемое правилом поддерево, то композиционные метки, расставляемые в дереве другими правилами множества Π_{LSC}^{R}, будут сохранены. Без изменения также остаются соответствующие элементы списков (2.23) для деревьев из множеств $T_{\chi 1}^{LSC}$ и $T_{\chi 2}^{LSC}$. Применение списков (2.23) и композиционных меток (2.22) таким образом позволяет избежать полного просмотра ЛФ-синонимических множеств при определении возможности построения рассматриваемой Δ-грамматикой очередного дерева.

2.5. Пример построения образа сверхфразового единства для четырех простых распространенных предложений русского языка

Рассмотрим работу предложенного механизма распознавания сверхфразовых единств на примере высказывания из четырех простых распространенных предложений русского языка:

1) “*Лаборатория провела эксперименты по изучению условных рефлексов*”;

2) ”*Подопытными животными были собаки*”;

3) ” *Результаты экспериментов рассматривались в докладе на конференции*”;

4) ”*Ученый детально анализировал результаты проведенных опытов*”.

С целью более наглядной демонстрации применения основных идей настоящей главы исходные предложения построены на основе лексики, описанной в Толково-комбинаторном словаре современного русского языка [118].

Скобочное описание дерева глубинного синтаксиса первого предложения с использованием структур вида (2.16), представленное на рис. 2.3, *б*, уже было затронуто нами в разделе 2.3. Аналогичные описания глубинных синтаксических структур для второго, третьего и четвертого предложений представлены на рис. 2.6.

```
(
 (( Экспериментировать  A2  Oper1 )
    (V (несов_вид прош_вр изъявит_накл )) 0 nil
 )
   ((( Собака nil )( S (мн_ч)) 1 nil ))
   ((( Экспериментировать  //A2 )( S (мн_ч)) 2 nil )
    ((( Q nil ) nil 1 nil ))
    ((( Q nil ) nil 2 nil ))
   )
)
```

а

Рис. 2.6. Анализируемые деревья глубинного синтаксиса:

а – второго; б – третьего; в – четвертого предложения

```
(
 (( Рассматривать nil )
    (V (несов_вид прош_вр изъявит_накл )) 0 nil
 )
   ((( Q nil )( S nil ) 1 nil ))
   ((( Экспериментировать S0 Sres )( S (мн_ч)) 2 nil )
    ((( Экспериментировать S0 )( S (мн_ч)) 2 nil )
      ((( Q nil ) ( S nil ) 1 nil ))
      ((( Q nil ) ( S ( на nil )) 2 nil ))
      ((( Q nil ) ( S ( по nil )) 3 nil ))
    )
   )
   ((( Доклад nil )( S (в ед_ч)) 3 nil )
    ((( Конференция nil )( S ( на nil )) 2 nil ))
   )
)
```

б

```
(
 (( Рассматривать Syn )
    (V (несов_вид прош_вр изъявит_накл )) 0 nil
 )
   ((( Ученый nil )( S (ед_ч)) 1 nil ))
   ((( Экспериментировать S0 Sres )( S (мн_ч)) 2 nil )
    ((( Экспериментировать S0 Syn )( S (мн_ч)) 2 nil )
      ((( Q nil )( S nil ) 1 nil ))
      ((( Q nil )( S (на nil)) 2 nil ))
      ((( Q nil )( S (по nil)) 3 nil ))
      ((( Проведенный nil )( P (мн_ч)) 5 nil ))
    )
   )
   ((( Q nil )( S (в nil)) 3 nil ))
   ((( Рассматривать Magn[‘аспекты’] )( Adv nil ) 5 nil ))
)
```

в

Рис. 2.6. Продолжение

Определяя применимость лексических синонимических преобразований, описанных в [45, стр. 152-159] и отвечающих *Теореме 2.9*, для глубинных синтаксических структур исходных предложений формируем списки вида (2.23), представленные в *Таблице 2.3*.

Таблица 2.3

Применимость лексических синонимических преобразований для исходных предложений

№ предложения	Результат анализа применимости
1	((17 1 Экспериментировать))
2	((16 1 Экспериментировать))
3	((1 1 Рассматривать)(1 2 Экспериментировать))
4	((1 1 Рассматривать)(1 2 Экспериментировать))

К первому предложению применимо лексическое правило № 17 с обслуживающим его синтаксическим правилом № 6, [45, стр. 154]. Заметим, что условие применимости данного правила, касающееся грамматических характеристик ключевого слова (C_0 – глагол), уже заложено в соответствующий компонент списочного описания (2.16) вершины выходного дерева и представлено символьным обозначением глагола из *Таблицы 2.1*. Соответствующий указанному условию переход в сети Петри, моделирующей рассматриваемую систему правил, является безусловным (значение соответствующего ему выражения вида (2.6) принимается тождественно равным “*true*”).

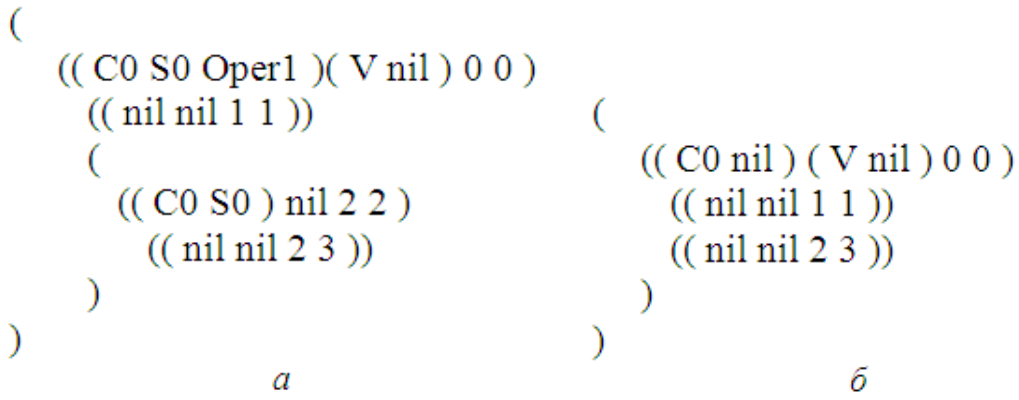


Рис. 2.7. Лексическое правило № 17 из представленных в [45, стр. 152-159]:

а – списочное описание входного; б – выходного дерева

Ко второму предложению применимо лексическое правило №16 с обслуживающим его синтаксическим правилом №8, [45, стр. 153]. Как и в предыдущем случае, условие применимости в виде логической формулы (2.6) отдельно не выносится и заложено в описании выходного дерева правила (рис. 2.8, *б*). Для обоих предложений лексико-синтаксические замены рассматриваются относительно ключевого слова C_0=*”Экспериментировать”*.

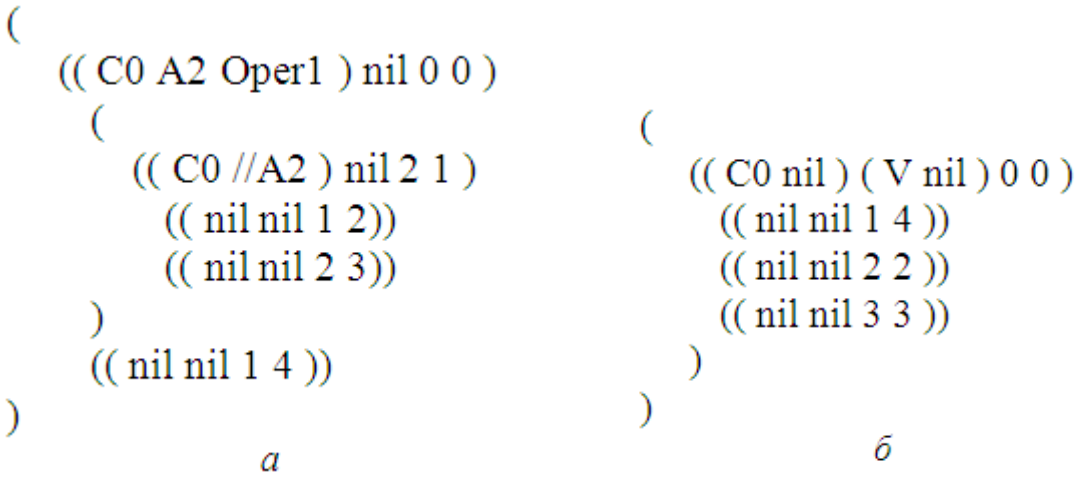


Рис. 2.8. Лексическое правило № 16 из представленных в [45, стр. 152-159]:

а – списочное описание входного; б – выходного дерева

Посредством применения указанных правил оба предложения приводятся к виду с одинаковой ЛСК относительно ключевого слова C_0="*Экспериментировать*".

1) "*Лаборатория экспериментировала на* S_{nn} *с целью изучения условных рефлексов*";
2) "*Экспериментировал (а,о,и) на собаках*".

Преобразованные деревья глубинного синтаксиса первого и второго предложения в скобочной нотации представлены на рис. 2.9.

```
(
 (( Экспериментировать nil )
    (V (несов_вид прош_вр изъявит_накл )) 0 nil
 )
   ((( Лаборатория nil )( S (ед_ч)) 1 nil ))
   ((( Q nil )( S (на мн_ч)) 2 nil ))
   ((( Изучение nil )( S (по ед_ч)) 3 nil )
    ((( Рефлекс nil )( S (мн_ч)) 2 nil )
       ((( Условный nil )( A (мн_ч)) 5 nil )
    )
   )
)
```

а

```
(
 (( Экспериментировать nil )
    (V (несов_вид прош_вр изъявит_накл )) 0 nil
 )
   ((( Q nil )( S (ед_ч)) 1 nil ))
   ((( Собака nil )( S (на мн_ч)) 2 nil ))
   ((( Q nil )( S (по ед_ч)) 3 nil ))
)
```

б

Рис. 2.9. Преобразованные деревья относительно C_0="*Экспериментировать*":

а – первого; б – второго предложения

Заполняя незаполненные места глубинно-синтаксических актантов в соответствии с *Определением 2.6*, получаем формальный образ сверхфразового единства для первого и второго предложений в виде ГСС на рис. 2.10.

```
(
 (( Экспериментировать  nil )
    (V (несов_вид прош_вр изъявит_накл )) 0 nil
 )
   ((( Лаборатория nil )( S (ед_ч)) 1 nil ))
   ((( Собака nil )( S (на мн_ч)) 2 nil ))
   ((( Изучение nil )( S (по ед_ч)) 3 nil )
    ((( Рефлекс nil )( S (мн_ч)) 2 nil )
      ((( Условный nil )( A (мн_ч)) 5 nil )
    )
   )
)
```

Рис. 2.10. Суммарная ГСС для первого и второго предложения

К дереву глубинного синтаксиса третьего предложения дважды применимо лексическое правило №1, [45, стр. 152], вход и выход которого в принятой нами скобочной нотации описывается как

(((C0 nil) nil 0 0)) и (((C0 Syn) nil 0 0)),

соответственно. Это же правило, но в обратном направлении, применимы к ГСС четвертого предложения. Посредством применения первого вхождения указанного правила относительно ключевого слова C_0=*"Рассматривать"* приводим ГСС обоих предложений к виду с одинаковой ЛСК. При этом дерево ГСС третьего предложения остается без изменений, а ГСС четвертого предложения приводится к виду, представленному на рис. 2.11.

```
(
 (( Рассматривать  nil )
    (V (несов_вид прош_вр изъявит_накл )) 0 nil
 )
   ((( Ученый nil )( S (ед_ч)) 1 nil ))
   ((( Экспериментировать  S0 Sres )( S (мн_ч)) 2 nil )
      ((( Экспериментировать  S0 Syn )( S (мн_ч)) 2 nil )
         ((( Q nil )( S nil ) 1 nil ))
         ((( Q nil )( S (на nil)) 2 nil ))
         ((( Q nil )( S (по nil)) 3 nil ))
         ((( Проведенный nil )( P (мн_ч)) 5 nil ))
      )
   )
   ((( Q nil )( S (в nil)) 3 nil ))
   ((( Рассматривать  Magn[‘аспекты’] )( Adv nil) 5 nil ))
)
```

Рис. 2.11. Преобразованное дерево четвертого предложения относительно C_0="*Рассматривать*"

Тем не менее, дерево глубинного синтаксиса третьего предложения (рис. 2.6, *б*) и преобразованная глубинная синтаксическая структура четвертого предложения (рис. 2.11) не могут функционально соответствовать друг другу по *Определению 2.8* в силу наличия синонима для C_0="*Экспериментировать*".

Указанное несоответствие устраняется применением второго правила из списка (2.23), представленного в *Таблице 2.3* для четвертого предложения. При этом дерево глубинного синтаксиса четвертого предложения преобразуется к виду на рис. 2.12. Формальный образ сверхфразового единства для третьего и четвертого предложения представлен на рис. 2.13.

```
(
 (( Рассматривать  nil )
    (V (несов_вид прош_вр изъявит_накл )) 0 nil
 )
   ((( Ученый nil )( S (ед_ч)) 1 nil ))
   ((( Экспериментировать  S0 Sres )( S (мн_ч)) 2 nil )
      ((( Экспериментировать  S0 )( S (мн_ч)) 2 nil )
         ((( Q nil )( S nil ) 1 nil ))
         ((( Q nil )( S (на nil)) 2 nil ))
         ((( Q nil )( S (по nil)) 3 nil ))
         ((( Проведенный nil )( P (мн_ч)) 5 nil ))
      )
   )
   ((( Q nil )( S (в nil)) 3 nil ))
   ((( Рассматривать  Magn[‘аспекты’]  )( Adv nil) 5 nil ))
)
```

Рис. 2.12. Окончательный вариант дерева четвертого предложения после замены синонима для C_0=*"Экспериментировать"*

```
(
 (( Рассматривать  nil )
    (V (несов_вид прош_вр изъявит_накл )) 0 nil
 )
   ((( Ученый nil )( S (ед_ч)) 1 nil ))
   ((( Экспериментировать  S0 Sres )( S (мн_ч)) 2 nil )
      ((( Экспериментировать  S0 )( S (мн_ч)) 2 nil )
         ((( Q nil )( S nil ) 1 nil ))
         ((( Q nil )( S (на nil)) 2 nil ))
         ((( Q nil )( S (по nil)) 3 nil ))
         ((( Проведенный nil )( P (мн_ч)) 5 nil ))
      )
   )
   ((( Доклад nil )( S (в ед_ч)) 3 nil ))
   ((( Рассматривать  Magn[‘аспекты’]  )( Adv nil) 5 nil ))
)
```

Рис. 2.13. Суммарная ГСС для третьего и четвертого предложения

Далее рассматриваем возможность суммирования деревьев глубинного синтаксиса, представленных на рис. 2.10 и 2.13. Аналогично деревьям дискретных предложений, определяем применимость лексических синонимических преобразований, описанных в [45, стр. 152-159] и отвечающих *Теореме 2.9*, для указанных глубинных синтаксических структур с формированием списков вида (2.23). Результаты представлены в *Таблице 2.4*.

Таблица 2.4

Применимость лексических синонимических преобразований для суммарных ГСС

№ предложений	Результат анализа применимости
1 и 2	((17 1 Экспериментировать))
3 и 4	((1 1 Рассматривать)(17 1 Рассматривать) (1 2 Экспериментировать))

Как видно из *Таблицы 2.4*, единственным ключевым словом, относительно которого возможно приведение суммарных ГСС к виду с одинаковой ЛСК, является C_0=*"Экспериментировать"*. Однако на основе начального сценария, соответствующего активизации входов/выходов лексических правил №1 и №17, требуемую последовательность преобразований в рассматриваемой системе правил построить нельзя. Поэтому предложенный механизм распознавания сверхфразовых единств для рассмотренного примера завершает свою работу, выдав в качестве окончательного результата деревья, представленные на рис. 2.10 и 2.13.

Выводы

Предложенный в настоящей главе подход к построению совокупности целевых выводов в Δ-грамматике позволяет теоретически обосновать принципиальную возможность существования алгоритмического решения для задач сравнения помеченных деревьев, требующих качественного анализа представленной в деревьях информации.

Применение данного подхода к задаче сжатия текстовой информации на уровне глубинного синтаксиса позволяет выделять семантические повторы в анализируемом тексте без существенного ограничения жанра анализируемых текстов, в то время как большинство из известных алгоритмически разрешимых методов распознавания сверхфразовых единств ориентированы на тексты определенного жанра.

Тем не менее, при практической реализации предложенного подхода актуальна проблема автоматизации накопления знаний об описываемых логическими формулами (2.6) условиях применимости правил синонимических преобразований глубинных синтаксических структур. В частности, требуется рассмотреть вопросы формализации толкования лексического значения слова, представляемого на естественном языке в специализированном толковом словаре, с целью автоматизированного получения и систематизации указанных знаний.

Решению данной задачи на основе идей и методов АФП посвящается третья глава работы.

Глава 3

СИТУАЦИИ СМЫСЛОВОЙ ЭКВИВАЛЕНТНОСТИ ТЕКСТОВ КАК ОСНОВА ФОРМИРОВАНИЯ ЗНАНИЙ О СИНОНИМИИ

Настоящая глава посвящена использованию семантически эквивалентных текстов в качестве исходных данных формирования и классификации семантических отношений как основы знаний о синонимии. На основе смысловых соотношений в рамках стандартных лексических функций дается понятие прецедента для ситуации ЛФ-синонимии. Решена задача автоматизации накопления знаний об условиях применимости правил синонимических преобразований деревьев глубинного синтаксиса. Предложено формализованное средствами логики предикатов первого порядка описание толкования лексического значения слова. Исследованы принципы обобщения независимых вариантов толкований слова относительно заданного предметно-ориентированного подмножества естественного языка. На основе методов АФП предложена и исследована модель системы элементов толкования, которые присутствуют в обобщаемых его вариантах.

3.1. Лексическое значение слова и его формализация на языке логики предикатов первого порядка

В рамках рассмотренного нами подхода "Смысл⇔Текст" большинство словарных единиц языка возникает при переходе от семантического представления к глубинным синтаксическим структурам. Фрагмент семантического представления, который соответствует отдельному ЕЯ-слову, представляет собой толкование

Лексического Значения (ЛЗ) этого слова. В работе [3] Ю.Д.Апресян исследует связь между толкованием слова и его МУ для решения задачи построения глубинной синтаксической структуры по фрагменту семантического представления. Цель настоящего раздела состоит в том, чтобы показать связь между толкованием ЛЗ слова и его смыслом, актуальную для формирования прецедента класса СЭ.

Как было уже показано нами в *Разделе 1.3*, прецеденту класса СЭ на верхнем уровне иерархии знаний о синонимии соответствует условие применимости некоторого правила синонимического преобразования глубинных синтаксических структур. Данное условие выполняет функцию фильтра, который запрещает синтез ЕЯ-фразы из множества семантически эквивалентных, если конечный продукт синтеза дает нарушение лексического значения, сочетаемости или стилистических норм. Многие фильтры были описаны в работах И.А. Мельчука, и А.К. Жолковского. Однако, как отметил академик Ю.Д.Апресян, проблема нуждается в дальнейшей разработке. Тем более, что по оценке И.А.Мельчука, специальных исследований по данному вопросу не проводилось, а сами правила описаны в первом приближении.

Следует отметить, что метод фильтров является традиционным методом построения синтаксической структуры фразы русского языка. Как показано в [72], его применение предполагает установление для большинства слов нескольких потенциально возможных связей с различными управляющими словами. Роль фильтров при этом состоит в выборе правильных вариантов анализа. Одним из подходов к решению задачи выбора корректного варианта здесь является привлечение семантической информации из словаря. Важнейшую роль при этом играет информация о семантической интерпретации глубинных синтаксических актантов предикатного

слова, описываемая его моделью управления. Тем не менее, при наличии у слова более одного ЛЗ становятся возможными альтернативные варианты разбиения анализируемой ЕЯ-фразы на словосочетания (Именные Группы (ИГ)), каждый из которых удовлетворяет требованию фильтров. В частности, для предикатных слов с каждым ЛЗ связывается альтернативный вариант МУ и соответствующий синоним с более широким, чем у самого слова, значением. При синонимическом преобразовании исходной ЕЯ-фразы на уровне глубинного синтаксиса названный фактор может привести к построению неадекватных перифраз.

Наиболее естественный путь решения показанной проблемы заключается в привлечении информации словарных определений (дефиниций, [3,93]) для тех понятий, которые обозначаются актантами предикатного слова. При этом введение в рассмотрение аналогичных определений для семантики произвольных отношений, отличных от связей предиката с актантами по МУ и задаваемых входящими в анализируемое предложение именными группами, позволяет более точно устанавливать соответствия требованиям семантической интерпретации глубинных синтаксических актантов предикатного слова при построении дерева ГСС.

Данная точка зрения естественным образом согласуется со сформулированным нами в *Разделе 1.4* определением прецедента класса СЭ. При этом исходными данными формирования условия применимости правила будут признаки слов в парах ЕЯ-высказываний, сравниваемых по смыслу. Далее в настоящей главе мы рассмотрим, каким образом данная информация может быть выявлена на основе лексикографического толкования слова.

В работе [93] на примере генитивной конструкции русского языка исследуется взаимодействие формальной и лексической

семантики в задаче построения формализованного описания значения слова. Представляемая Б.Х. Парти и В.Б.Борщевым идея состоит в выделении совокупности свойств обозначаемого словом объекта реального мира и последующим описанием ЛЗ слова посредством теории – совокупности аксиом (meaning postulates), каждая из которых описывает отдельное свойство этого объекта. Само задаваемое посредством набора аксиом описание ЛЗ слова здесь соответствует теории сорта обозначаемой словом реалии. При этом понятие сорта как элемента "наивной картины мира" и класса, к которому язык относит конкретную реалию, фактически соответствует тому, что в публикациях Московской лингвистической школы, в частности, в монографиях [3] и [45], понимается под Семантическим Классом (СК) обозначающего эту реалию слова. Такое же понимание СК использовалось нами и в [21] относительно описания семантической интерпретации глубинного синтаксического актанта предикатного слова. Для описания самой теории сорта в [93] используется принятое в формальной семантике λ-выражение (выражение с оператором абстракции лямбда [6], которое возвращает в качестве значения множество всех объектов, принадлежащих заданному сорту.

Рассмотрим вначале ряд свойств формализованного описания ЛЗ в виде теории, предложенного в [93], которые необходимо принять во внимание при программной реализации соответствующего компонента словарной базы знаний.

Во-первых, предлагаемая в [93] теория для сорта опорного существительного именной группы есть описание свойств объектов, принадлежащих данному сорту. Фактически это означает, что из всех возможных отношений, задаваемых именными группами и связываемых с лексическими значениями их опорных слов,

первоочередную значимость для нас имеют лексические отношения – те отношения, которые задаются самими опорными словами.

Во-вторых, оператор типового сдвига для преобразования унарных отношений типа $\langle e,t \rangle$[3], которые исходно сопоставляются словарным значениям опорных слов именных групп, в задаваемые этими ИГ бинарные отношения (пример – метонимический сдвиг слова с ЛЗ "*контейнер*" в сорт "*квант*", описанный в [93]). Введение такого оператора требует формального описания уже не теории сорта, а задаваемого этим сортом отношения. При этом и имя отношения (как имя сорта), и его аргументы представляются аргументами функции – λ-выражения, сопоставляемого именной группе. Здесь следует отметить, что имя отношения, определяемого сортом опорного слова ИГ, как и сам этот сорт, в терминологии Московской лингвистической школы следует отождествлять с семантическим классом, но не отдельного слова, а всего словосочетания именной группы. Так, для глагола “*сжечь*” в значении “*израсходовать*” семантический класс актанта количественной ролевой ориентации ("*Quant*") соответствует именно количественному отношению ("*Quant*", "*квант*"), которое задается рассмотренной в [93] генитивной конструкцией меры (пример: “*сжечь машину дров*”).

В-третьих, в концептуальном плане теория лексического значения слова представляется набором утверждений, связывающих его с другими словами (в первую очередь здесь рассматривается связь между обозначаемыми словами понятиями). Отдельное утверждение теории описывает бинарное отношение между некоторыми известными понятиями. Каждое из понятий, выступающее в роли

[3] Здесь имеется в виду используемое в формальной семантике понятие типа, *e* и *t* соответствуют элементарным типам: сущностям и формулам.

аргументов отношения, по сути, соответствует одному из известных СК. Имя самого отношения задается ЕЯ-словом, для которого явным образом в словарной базе знаний указан семантический класс обозначаемой этим словом сущности.

В работе [93] в качестве аргументов функции, описывающей задаваемое генитивной конструкцией отношение, выступают элементы конструкции – опорное слово и генитивная группа (зависимое слово). Но, рассуждая о приемлемости той или иной генитивной конструкции, принято говорить не о входящих в нее словах, а о сортах обозначаемых этими словами реалий. Исходя из этого соображения, в настоящей работе теорию отношения, определяемого ИГ, мы будем рассматривать не относительно самих слов-элементов именной группы, а относительно соответствующих им семантических классов.

На основе вышесказанного, а также в соответствии со сформулированной нами *Задачей 1.2*, представим описание теории ЛЗ слова w_i, заменяемого посредством некоторого правила $\pi \in \Pi^R$, в виде структуры:

$$Lm(w_i) = \left(w_i, L^M\right), \tag{3.1}$$

где L^M – список структур, задающих отношения между словами и понятиями. При этом элементом списка L^M может быть как бинарное отношение между парой понятий C_1 и C_2:

$$M_p = (R_2, C_1, C_2), \tag{3.2}$$

так и рекурсивно определяемое отношение произвольной арности вида

$$M'_p = \left(R_n, C, L^M\right), \tag{3.3}$$

либо

$$M''_p = \left(R_c, L^M\right), \qquad (3.4)$$

где $R_c \in \{\vee, \&, \neg\}$. Посредством L^M в (3.3) задается связь понятия C с другими словами и понятиями.

Сама теория ЛЗ слова, задаваемая структурой вида (3.1), может быть представлена составным объектом языка Пролог, в свою очередь легко преобразуемым в структуры специализированного домена *tree* для работы с деревьями в Visual Prolog'е.

На рис. 3.1 приведено древовидные описания теорий для ЛЗ слов "*эксперимент*" и "*экспериментировать*", упоминавшихся в примере из *Раздела 2.5*. Исходные варианты толкований взяты из Толково-комбинаторного словаря современного русского языка И.А. Мельчука и А.К. Жолковского [118].

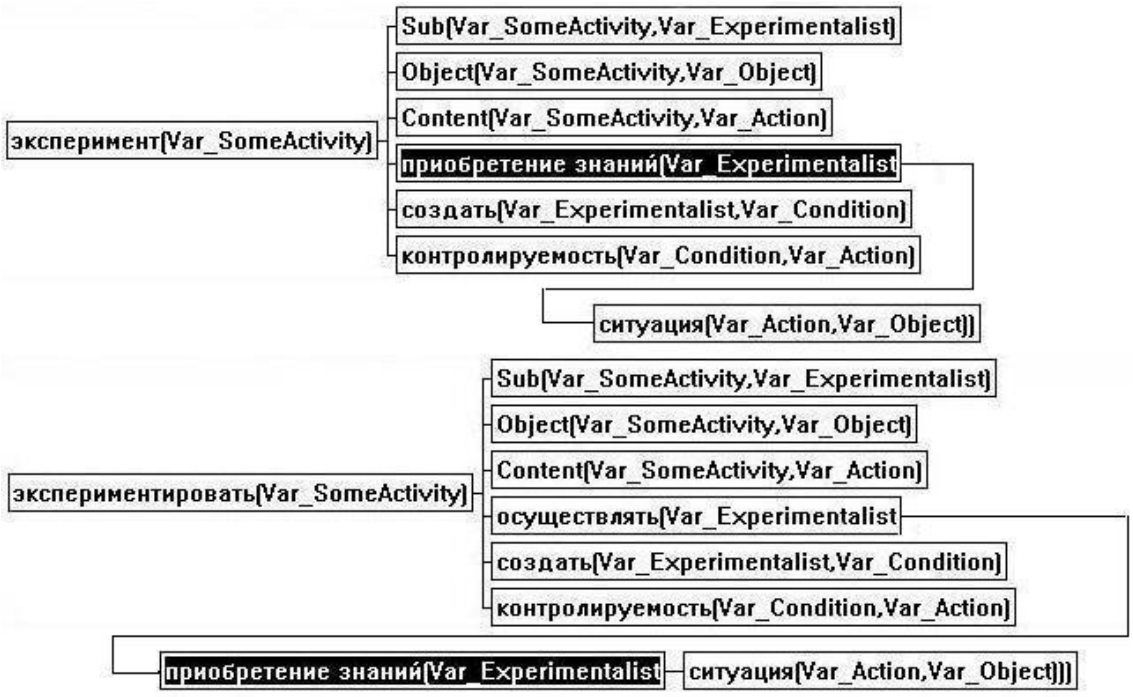


Рис. 3.1. Теории ЛЗ "эксперимент" и "экспериментировать"

Утверждение 3.1. Если имеется формализованное описание теории $Lm(w_i) = \left(w_i, L^M\right)$ ЛЗ слова w_i, задаваемое структурой вида (3.1), то смысл слова этого слова определяется набором Характеристических Функций (ХФ) ChF_{hi} таких, что выполняются следующие условия:

1. В списке L^M содержится структура $M_p = (R_2, C_1, C_2)$ вида (3.2) (обозначим ее как ChF_{Val}), при этом $ChF_{hi}(w_i) = C_2$, где C_2 – обозначение известного системе понятия (семантического класса), а сам список L^M может быть третьим аргументом структуры (3.3).
2. Существует структура (далее обозначаемая как ChF_{Name}) либо вида (3.2) и при этом $M_p = \left(ChF_{hi}, C_1^1, C_2^1\right)$, либо вида (3.3) и при этом $M'_p = \left(ChF_{hi}, C, L^M\right)$, но в обоих случаях ChF_{hi} – имя известного смыслового (семантического) отношения.
3. Если ChF_{Name} есть первая структура, удовлетворяющая условию (2) при обратном просмотре списка L^M от ChF_{Val}, и $L^{M'} \subset L^M$ есть список такой, что либо $L^{M'} = \left\{\left(ChF_{hi}, C_1^1, C_2^1\right), \ldots, ChF_{Val}\right\}$, либо $L^{M'} = \left\{\left(ChF_{hi}, C, L^M\right) : ChF_{Val} \in L^M\right\}$, то каждое последующее утверждение в $L^{M'}$ должно иметь как минимум один общий аргумент, являющийся обозначением некоторой переменной, с предыдущим утверждением.

В качестве примера на рис. 3.2 представлен вариант теории для ЛЗ слова “*агрессор*”, а на рис. 3.3 – соответствующий ему набор характеристических функций. Как и для примера на рис. 3.1, исходный вариант толкования взят в [118].

Рис. 3.2. Анализируемый вариант теории ЛЗ

Рис. 3.3. Характеристические функции и формальные признаки их значений

Здесь *Var_SomeBody* обозначает переменную для слова, интерпретируемого посредством структуры (3.1). Эта же переменная является вторым аргументом для ChF_{Name} согласно введенным обозначениям в *Утверждении 3.1*.

Фактически посредством *Утверждения 3.1* мы сформулировали точное определение смысла слова на основе *Определения 1.6*, более близкого пониманию смысла в философской логике. Опираясь на понятия экстенсионала и интенсионала, рассмотрим решение задачи обобщения знаний, представляемых структурами вида (3.1), на основе математических методов АФП. Данная задача актуальна при независимом построении теории слова разными исследователями, в частности, при построении теорий на основе ЕЯ-толкований с применением стандартных концептуальных языков [19,85,86,128].

Представим систему элементов толкования заданного слова для независимых вариантов теории лексического значения посредством многозначного формального контекста вида

$$K^{LM} = \left(G^{LM}, M^{LM}, V^{LM}, I^{LM}\right), \qquad (3.5)$$

где $\forall g^{LM} \in G^{LM}$ есть некоторый вариант толкования ЛЗ слова w_i в форме (3.1). Множество признаков $M^{LM} = M_1^{LM} \cup M_2^{LM}$, при этом если $m^{LM} \in M_1^{LM}$, то $m^{LM} = ChF_{hi}(w_i)$, а если $m^{LM} \in M_2^{LM}$, то m^{LM} – это имя некоторого известного семантического класса или отношения, выступающего к качестве первого аргумента структуры вида (3.2) в составе списка $L^{M'} \subset L^M$, формируемого в соответствии с условием (3) *Утверждения 3.1*. Множество признаковых значений $V^{LM} = V_1^{LM} \cup V_2^{LM}$, при этом если $v^{LM} \in V_1^{LM}$, то v^{LM} есть имя ХФ ChF_{hi}, для которой задано $ChF_{hi}(w_i)$. Если же $v^{LM} \in V_2^{LM}$, то v^{LM} есть значение ХФ $ChF'_{hi}(w_i^1)$: $Lm\left(w_i^1\right) = \left(w_i^1, L^{M'}\right)$. Тернарное отношение I^{LM} задает частичное отображение G^{LM} на V^{LM}: $m^{LM}\left(g^{LM}\right) = v^{LM}$, ставит в соответствие каждой ХФ ее значение для заданного w_i.

Лексическое значение слова, описываемое посредством формализованной теории (3.1), есть денотация. В логике ей ставится в соответствие экстенсионал как класс сущностей, которые определяются посредством теории. При этом внешне различные описания теорий одного и того же ЛЗ определяют единое множество характеристических функций, задаваемых в соответствии с *Утверждением 3.1*. Характеристические функции (в том числе определяемые рекурсивно для списочных аргументов структур (3.3) и (3.4)), задают набор формальных признаков для элементов толкования

лексического значения. В конечном итоге они определяют интенсионал обобщенной теории заданного лексического значения.

Таким образом, исходя из определения интенсионала как функции от возможных миров к экстенсионалам, а также рекурсивной природы постулатов значения, имеем *задачу* построения обобщенной теории лексического значения как восстановление синтаксического представления экстенсионала на основе известного синтаксиса λ-выражений для характеристических функций, составляющих интенсионал.

Утверждение 3.2. Утверждения $\left(R_n, C, L_1^M\right)$ и $\left(R_n, C, L_2^M\right)$ вида (3.3) могут быть представлены одним “ИЛИ”-утверждением:

$$\left(R_n, C, \left\{\left\langle "or", L_3^M\right\rangle\right\}\right),$$

если наборы ФП, полученные на основе L_1^M, L_2^M и L_3^M образуют области $\Re^{LM}\left(G_1^{LM}, M_1^{LM}, V_1^{LM}, I^{LM}\right)$, $\Re^{LM}\left(G_2^{LM}, M_2^{LM}, V_1^{LM}, I^{LM}\right)$ и, соответственно, $\Re^{LM}\left(G_3^{LM}, M_3^{LM}, V_1^{LM}, I^{LM}\right)$ с НОСП, которое имеет R_n в качестве значения признака. При этом:

$$G_1^{LM} = \left\{\left\langle w_i^1, L_1^M\right\rangle\right\},\ G_2^{LM} = \left\{\left\langle w_i^2, L_2^M\right\rangle\right\},\ M_1^{LM} \neq M_2^{LM},\ M_3^{LM} = M_1^{LM} \cup M_2^{LM},$$

$$\Re^{LM}\left(G_3^{LM}, M_3^{LM}, V_1^{LM}, I^{LM}\right) =$$

$$= \Re^{LM}\left(G_1^{LM}, M_1^{LM}, V_1^{LM}, I^{LM}\right) \cup \Re^{LM}\left(G_2^{LM}, M_2^{LM}, V_1^{LM}, I^{LM}\right).$$

Утверждение 3.3. Утверждения $\left(R_n, C, L_1^M\right)$ и $\left(R_n, C, L_2^M\right)$ вида (3.3) могут быть представлены одним “И”-утверждением:

$$\left(R_n, C, \left\{\left\langle "and", L_3^M\right\rangle\right\}\right),$$

если на основе L_1^M, L_2^M и L_3^M определяются ФП (X, Y_1), (X, Y_2) и (X, Y_3): $Y_3 = Y_1 \cup Y_2$.

Замечание. Согласно *Утверждению 3.1*, внешне различные описания теорий вида (3.1) для одного и того же ЛЗ задают единое множество характеристических функций. Следовательно, мощность указанного множества не зависит от количества обобщаемых теорий. Временная оценка процесса обобщения теорий для заданного ЛЗ составляет $\left(\frac{n}{k}\right)^k$, где n – мощность множества ХФ, k - количество обобщаемых теорий. Поскольку $k \in [1,\ldots,n]$, то $\left(\frac{n}{k}\right)^k = n$ при $k = 1$ и $\left(\frac{n}{k}\right)^k = 1$ при $k = n$. В худшем случае n равно числу утверждений вида (3.2) и (3.3) на всех уровнях описания ЛЗ структурой (3.1).

В качестве примера на рис. 3.4 представлена решетка ФП для трех вариантов толкования ЛЗ “*агрессор*”, а на рис. 3.5 –результат их обобщения. Помимо ТКС [118], исходные варианты толкования были взяты из Большой Советской Энциклопедии, тематического словаря "Война и мир" и словаря Брокгауза и Ефрона [92]. В настоящей главе (кроме *Раздела 3.5*) для визуализации решеток диаграммами линий используется специализированный программный продукт ToscanaJ [127], реализующий методы АФП.

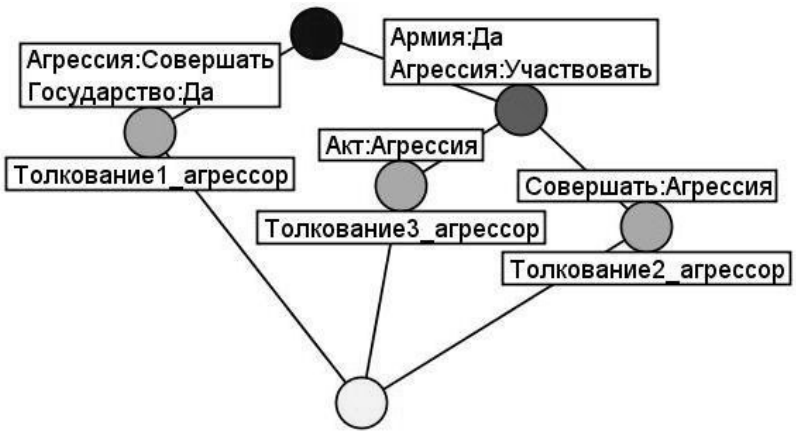


Рис. 3.4. Формализованные толкования для ЛЗ "*агрессор*"

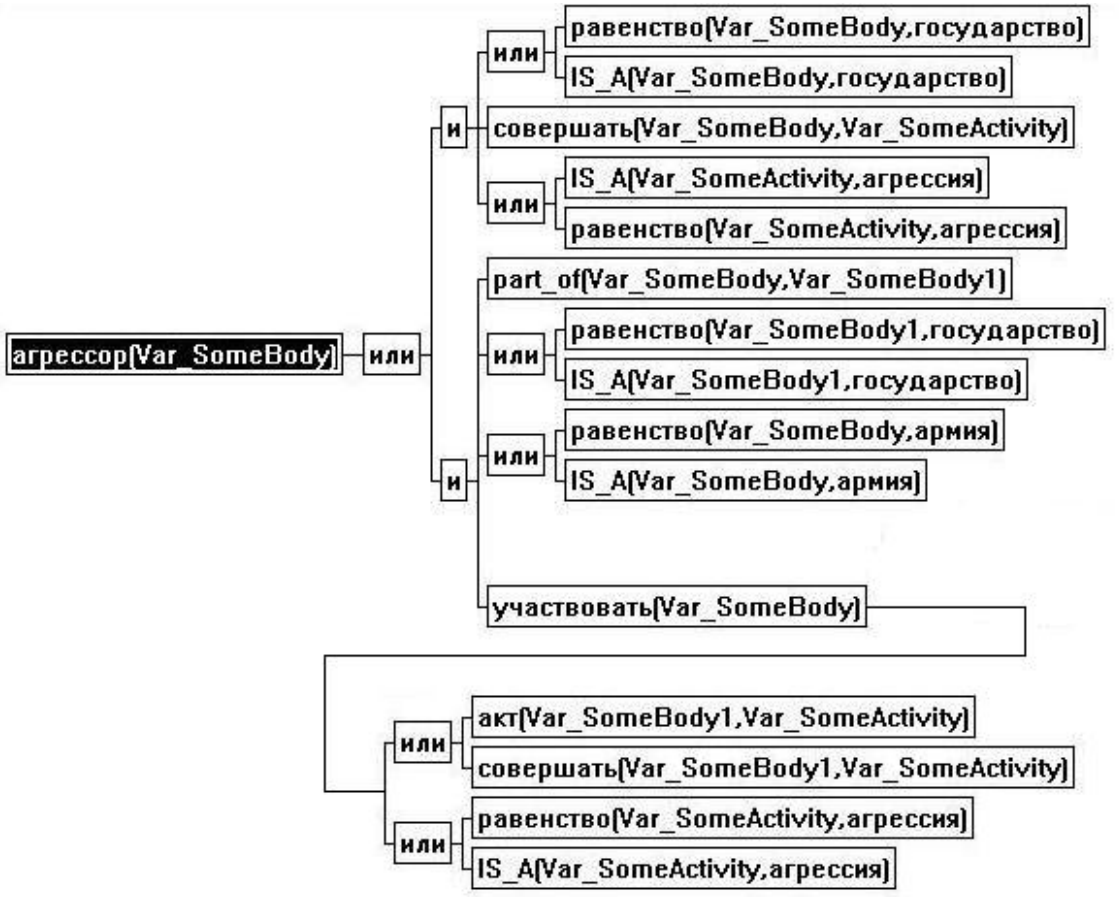


Рис. 3.5. Обобщенная теория ЛЗ "*агрессор*"

Ключевое правило обобщения утверждений независимых вариантов теории лексического значения определяется введением в рассмотрение области, которую образуют элементы толкования заданного лексического значения в решетке формальных понятий. Это позволяет различать случаи:

– использования разных ХФ с одним и тем же значением в независимых альтернативных вариантах теории ЛЗ (обобщение посредством отношения "ИЛИ", рис. 3.5). В формальном контексте на рис. 3.4 примерами являются пара ФП ("*Толкование2_агрессор*", "*Толкование3_агрессор*") и пара, образованная "*Толкованием1_агрессор*" и НОСП для пары ("*Толкование2_агрессор*", "*Толкование3_агрессор*");

– описания одного и того же элемента толкования ЛЗ, но посредством разных ХФ (обобщение посредством отношения "И", рис. 3.5). В представленном на рис. 3.4 формальном контексте примером может послужить содержание ФП "*Толкование1_агрессор*", а также содержание НОСП для пары ("*Толкование2_агрессор*", "*Толкование3_агрессор*").

При этом вычислительная сложность процесса обобщения теорий заданного ЛЗ зависит исключительно от мощности множества характеристических функций. Согласно определению смысла как интенсионала лексического значения, число самих ХФ не зависит от количества обобщаемых теорий. В перспективе для утверждений, объединяемых посредством отношения "ИЛИ", здесь появляется возможность задействования статистических методов для выявления наиболее значимых признаков.

3.2. Прецеденты семантических отношений для ситуаций синонимии на основе стандартных лексических функций

При формировании прецедентов СЭ для ситуаций использования лексических функций-параметров *актуальна задача* выявления и обобщения смыслового отношения в рамках расщепленного значения. В настоящем разделе мы рассмотрим, каким образом данная задача может быть решена с привлечением информации ЛЗ, формализуемого посредством теорий вида (3.1).

Пусть $r(\pi)$ условие применимости правила $\pi \in \Pi^R$, W_1 и W_2 – комплексы лексических единиц, заменяемых посредством π согласно постановке *Задачи 1.2*, а $W = W_1 \cup W_2$.

Определение 3.1. Пара (W_1, W_2) соответствует *Расщепленнму Значению (РЗ)* при обязательном выполнении следующих условий:

1. $\forall w_i \in W_1$ либо является значением некоторой лексической функции для ключевого слова C_0, определяющего ситуацию СЭ, либо есть само C_0.
2. $\exists w_k \in W_1$: $w_k = F(C_0)$ и F относится к классу лексических функций-параметров [45, стр. 78].
3. $W_2 = \{w\}$, при этом w есть либо значение некоторой ЛФ-замены [45, стр. 78] для данного C_0, либо есть само C_0. Комплекс W_2 соответствует нерасщепленному смысловому эквиваленту расщепленного значения, отождествляемого с W_1.

Заметим, что актуальное для формализации $r(\pi)$ перераспределение смысла между лексемами характерно для ситуаций с ЛФ-параметрами. В общем случае формирование прецедента для ситуации СЭ на основе РЗ предполагает наряду с формализацией требований к смыслу слов в составе каждого W_j, $j \in \{1,2\}$, выявление и обобщение смыслового отношения между произвольными w_i и w_m, входящими в W и отвечающими нижеперечисленным требованиям:

1. $w_i \neq w_m$.
2. w_i, есть значение некоторой лексической функции-параметра для заданного C_0.
3. w_m есть либо значение некоторой лексической функции-замены для заданного C_0, либо $w_m = C_0$.

Пример. РЗ *“осуществлять эксперимент”*, где значением ЛФ $Oper_1$ задается смысловое отношение наподобие *“операция с”* между

1-м участником ситуации СЭ (кто осуществляет эксперимент) и ее названием (“*эксперимент*”). Данное РЗ имеет нерасщепленный эквивалент “*экспериментировать*”.

Такис образом, требования к заменяемым лексическим единицам, предъявляемые условием $r(\pi)$, определяются смысловыми отношениями между ключевым словом C_0 и его лексическими коррелятами, которые входят в заменяемый комплекс лексических единиц. В лексической семантике именно такие отношения и описываются стандартными лексическими функциями. Фактически для ситуации СЭ на основе расщепления лексического значения расщепленное значение определяет этот вид отношений. Указанный факт позволяет поставить задачу выявления и обобщения смыслового отношения в рамках РЗ по аналогии с описанием семантики именных групп на основе формализованного представления толкований лексических значений слов в виде теорий (3.1).

Сказанное подтверждается наработками по Русскому общесемантическому словарю: лексические функции используются в качестве Семантических Характеристик (СХ) отдельных слов в РОСС. А это означает, что такие слова могут быть и названиями отношений в утверждениях теорий других слов. Примером может послужить значение ЛФ $Oper_1$ для ЛЗ “*эксперимент*” (т.е. “*осуществлять*”, рис. 3.1), которое присутствует в одном из утверждений теории ЛЗ “*экспериментировать*”. При этом применение лексических функций в качестве СХ отдельных слов в указанном словаре позволяет сделать вывод о возможности выявления смысловых зависимостей, определяемых лексическими функциями, путем сравнительного анализа множеств аксиом теорий ЛЗ слов в расщепленном значении.

Утверждение 3.4. Смысловое отношение F, значимое для формирования $r(\pi)$, между некоторым словом w_2 и его лексическим коррелятом w_1, входящим в РЗ, будет иметь место тогда, когда

$$L_1^M = L_{11}^M \cup L_{22}^M \cup L_{12}^M,$$

$$L_2^M = L_{11}^M \cup \{\langle F, C, L_{22}^M \rangle\} \cup L_{12}^M,$$

$$L_{11}^M \cap L_{22}^M = \varnothing,\ L_{11}^M \cap L_{12}^M = \varnothing,\ L_{12}^M \cap L_{22}^M = \varnothing,$$

где L_1^M – набор утверждений теории ЛЗ для w_1; L_2^M – аналогичный набор для w_2.

При независимом построении теорий для одного и того же слова, но разными исследователями и на основе разных корпусов текстов, возникает задача контроля адекватности и полноты сочестаемости слова по заданной ЛФ. В следующем разделе мы покажем, каким образом данная задача может быть решена совместным использованием информации моделей управления предикатных слов и формализованных теорий лексических значений.

3.3. Семантика расщепленного значения и смысловые валентности предикатного слова

В докладе [59] нами было рассмотрено использование семантической информации предикатных слов русского языка, представленной в Русском общесемантическом словаре, для безошибочной идентификации отношения "более общее – более частное" (в терминологии АФП – "подпонятие-суперпонятие") между предикатными словами на основе анализа ролевого состава их ЛЗ. Следует отметить, что описание дифференциальных признаков слова цепочками СХ в указанном словаре есть разновидность формульного

описания, представимого структурой (3.1), для теории СК этого слова. Каждая СХ соответствует некоторой "семантической координате" (сорту, [93]) обозначаемой словом сущности. К настоящему моменту идеология РОСС имеет практическое воплощение в разработанном рабочей группой Aot.ru АРМ лингвиста [2].

Использование лексических функций в качестве СХ отдельных слов в РОСС позволяет сделать вывод об использовании таких слов в качестве названий отношений в утверждениях теорий других слов а, следовательно, и возможности выявления смысловых зависимостей, определяемых лексическими функциями, путем сравнительного анализа множеств аксиом теорий ЛЗ слов в расщепленном значении. Согласно *Утверждению 3.4*, сравнение производится на предмет наличия зависимости, определяемой семантическим отношением в некотором постулате вида (3.2) или (3.3) одной из сопоставляемых теорий. При этом подмножество аксиом теории ЛЗ другого слова либо является одним из аргументов этого отношения, либо непосредственно задается одним из сравниваемых слов. Примером могут послужить теории ЛЗ "*эксперимент*" и "*экспериментировать*", представленные на рис. 3.1.

Лексическими функциями описывается в первую очередь лексическая сочетаемость, которая определяется лексическим значением ключевого слова ЛФ-синонимической замены. Следовательно, ЛЗ более узкого по смыслу слова (в терминологии АФП – гипонима) включает лексические значения более широких по смыслу слов (гиперонимов), которые упоминаются в толковании ЛЗ рассматриваемого слова, а, следовательно, и в его теории. Таким образом, слово-гипоним в большинстве случаев будет иметь в качестве значений ЛФ-параметра значения этой же ЛФ для тех слов-гиперонимов, которые упоминаются в его толковании (теории).

Сказанное позволяет для заданной ЛФ описать систему слов, являющихся ее аргументами, посредством формального контекста:

$$K^{LF} = \left(G^{LF}, M^{LF}, I^{LF}\right), \qquad (3.6)$$

где множество объектов G^{LF} есть множество ключевых слов-аргументов заданной лексической функции. Множеству формальных признаков M^{LF} соответствует множество слов-значений лексической функции для слов из G^{LF}. Бинарное отношение $I^{LF} \subseteq G^{LF} \times M^{LF}$ задает частичное отображение G^{LF} на M^{LF} и ставит в соответствие каждому ключевому слову $C_0 \in G^{LF}$, определяющему ситуацию СЭ, множество значений заданной лексической функции. В качестве примера на рис. 3.6 представлена модель вида (3.6) для слов-аргументов ЛФ $Oper_1$ из верхней окрестности для ЛЗ *“эксперимент”*.

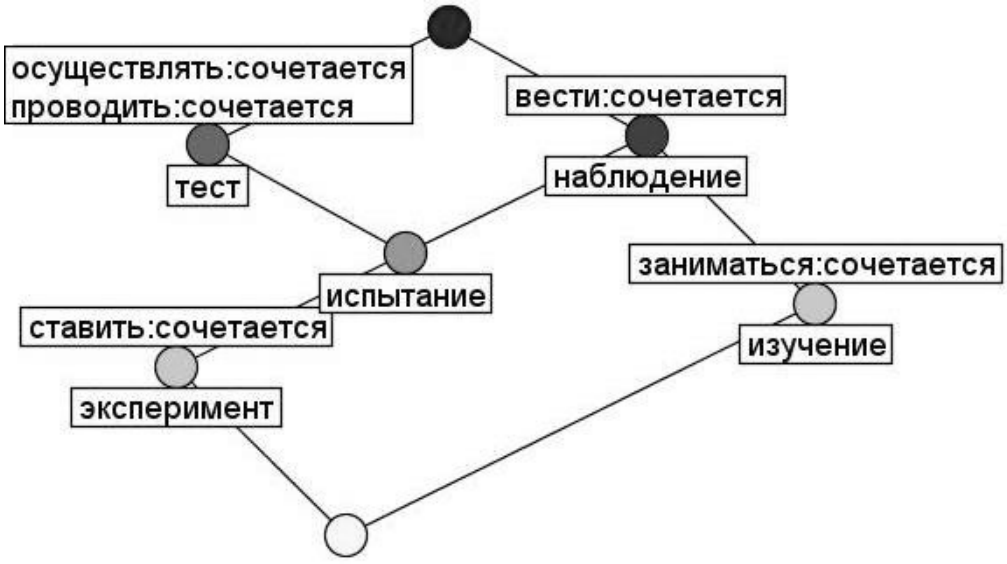


Рис. 3.6. Слова-аргументы лексической функции $Oper_1$ из верхней окрестности для лексического значения *“эксперимент”*

С другой стороны, для предикатных слов отношение “гипоним-гипероним” определяется, как было показано нами в [59], в первую

очередь анализом смысловых валентностей. Поэтому для оценки адекватности классификации объектов множества G^{LF} на основе формального контекста (3.6) рассмотрим определение отношения гипонимии между семантическими классами с учетом формульных описаний вида (3.1) для семантических характеристик слова.

Пусть для каждого слова w_i мы имеем описание его семантического класса C_i^{SF} посредством четверки:

$$S_i^{SF} = \left(C_i^{SF}, L_i^{SF}, D_i^{SF}, D_{ij}^{SF}\right), \tag{3.7}$$

где второй, третий и четвертый элементы указывают на дескрипторы, используемые в РОСС для однозначной идентификации C_i^{SF}. При этом компонент L_i^{SF} есть список дескрипторов семантических характеристик в последовательности “более общая СХ – более специфическая СХ”. Дескрипторы D_i^{SF} и D_{ij}^{SF} обозначают таксономическую категорию и ее подкласс, соответственно.

Предположим также, что w_i есть предикатное слово. При этом для его семантического класса имеется описание характеризованного ролевого состава:

$$C^A = \left(C_i^{SF}, L_i^R\right), \tag{3.8}$$

где $\forall A_{ti} \in L_i^R$ включает название R_{ti} роли плюс список L_{ti}^C возможных семантических классов актанта:

$$A_{ti} = \left(R_{ti}, L_{ti}^C\right). \tag{3.9}$$

Утверждение 3.5. ЛЗ слова, относящегося к СК C_1^{SF}:

$$C_1^A = \left(C_1^{SF}, L_1^R\right)$$

следует считать суперпонятием для ЛЗ слова СК C_2^{SF}:

$$C_2^A = \left(C_2^{SF}, L_2^R\right)$$

тогда, когда для $\forall R_t: \left(R_t, L_{t2}^C\right) \in L_2^R \ \exists\left(R_t, L_{t1}^C\right) \in L_1^R$ такой, что каждому $C_{at1} \in L_{t1}^C$ можно поставить в соответствие $C_{at2} \in L_{t2}^C$, который либо равен C_{at1}, либо связан с C_{at1} отношением “вид-род”.

Утверждение 3.6. ЛЗ слова w_i, относящегося к СК C_i^{SF}:

$$S_i^{SF} = \left(C_i^{SF}, L_i^{SF}, D_i^{SF}, D_{ij}^{SF}\right)$$

следует считать суперпонятием для ЛЗ слова w_m СК C_m:

$$S_m^{SF} = \left(C_m^{SF}, L_m^{SF}, D_i^{SF}, D_{ij}^{SF}\right),$$

если в дополнение к определенным *Утверждением 3.5* условиям при отсутствии для актанта $A_{ai} = \left(R_{ai}, L_{ai}^C\right): A_{ai} \in L_i^R$, описываемого в составе структуры (3.8), актанта подпонятия с показанным в *Утверждении 3.5* соответствием набора возможных СК существует актант $A_{bm} = \left(R_{bm}, L_{bm}^C\right): A_{bm} \in L_m^R$, отвечающий нижеследующему требованию. Пусть для $\forall C_{qai}^{SF} \in L_{ai}^C$ задано описание

$$S_{qai}^{SF} = \left(C_{qai}^{SF}, L_{qai}^{SF}, D_{qai}^{SF}, D_{qai1}^{SF}\right)$$

и аналогично для $\forall C_{sbm}^{SF} \in L_{bm}^C$

$$S_{sbm}^{SF} = \left(C_{sbm}^{SF}, L_{sbm}^{SF}, D_{sbm}^{SF}, D_{sbm2}^{SF}\right).$$

Тогда наряду с вхождением в список L_{sbm}^{SF} семантических характеристик из списка L_{qai}^{SF} некоторым СХ $SF_{pqai} \in L_{qai}^{SF}$ ставятся в соответствие формализованные описания (3.1):

$$Lm_{pqai} = (SF_{pqai}, L_{pqai}^M),$$

причем $\exists L_{sbm}^{SF'} \subset L_{sbm}^{SF}: \forall SF_{osbm} \in L_{sbm}^{SF'}$ является в составе L_{pqai}^{M} либо одним из аргументов структуры (3.2), либо первым аргументом структуры (3.3).

Примером указанного соответствия может послужить аспектная валентность у ЛЗ *"испытание"* и валентность содержания у ЛЗ *"тест"* из представленных на рис. 3.6 слов верхней окрестности ЛЗ *"эксперимент"*.

Действительно, согласно указанному в *Утверждении 3.5* условию существования отношения гипонимии между лексическими значениями, ЛЗ *"тест"* не может выступать в качестве суперпонятия для ЛЗ *"испытание"*. Основание – отсутствие задаваемого *Утверждением 3.5* соответствия для валентности аспекта у ЛЗ *"испытание"* и валентности содержания у ЛЗ *"тест"*. Тем не менее, в словарной базе данных АРМ лингвиста [2] для семантического класса слова, реализующего аспектную валентность у ЛЗ *"испытание"* и для семантического класса слова, реализующего валентность содержания у ЛЗ *"тест"*, представлены описания совокупностями вышеупомянутых дескрипторов семантических характеристик, таксономических категорий и их подклассов.

Имеем:

$w_i =$ ”*тест*”, $w_m =$ ”*испытание*”,

$S_{qai}^{SF} =$ (”*ситуация*”, [”SITUAT”], ”LABL”, “SIT”),

$S_{sbm}^{SF} =$ (”*свойство*”, [”ATTR”], “ASP”, “*Не определена*”).

Кроме того, имеем также теорию сорта, отождествляемого с СХ "SITUAT" (рис. 3.7).

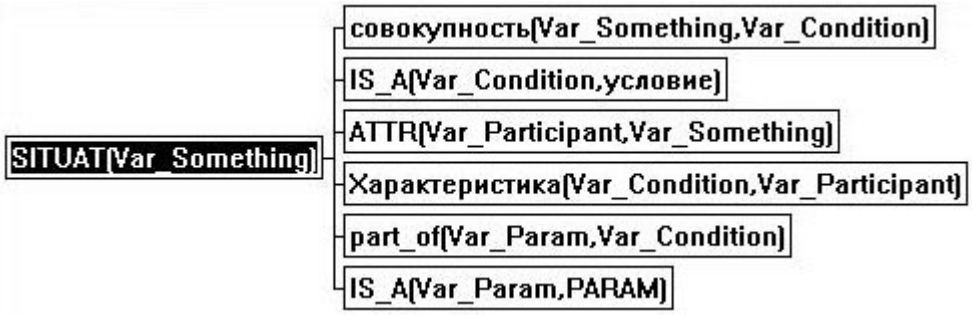


Рис. 3.7. Теория сорта "SITUAT"

Как видно из приведенного на рис. 3.7 древовидного описания, теория сорта "SITUAT", упоминаемого в списке СХ для ЛЗ *"ситуация"*, "ссылается" на семантические характеристики "ATTR" и "PARAM", из которых "ATTR" присутствует в списке СХ для ЛЗ *"свойство"*. Таким образом, относительно ЛЗ *"испытание"*, ЛЗ *"тест"* удовлетворяет сформулированным нами требованиям к суперпонятию лексического значения.

Визуализируя (рис. 3.8) средствами Visual Prolog'a отношение гипонимии для множества СК слов-аргументов заданной ЛФ, мы можем оценить как адекватность и полноту описания слова по ЛФ, так и корректность лексикографического толкования как основы для построения модели управления этого слова (рис. 3.9).

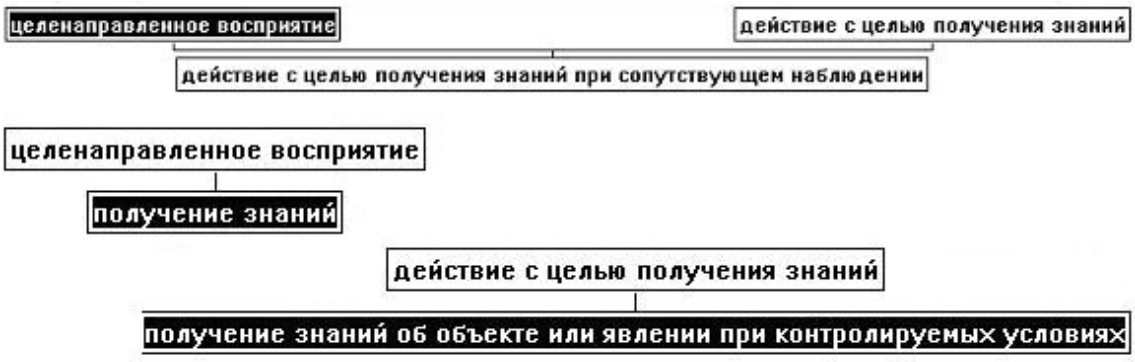


Рис. 3.8. Семантические классы слов окрестности ЛЗ "*эксперимент*"

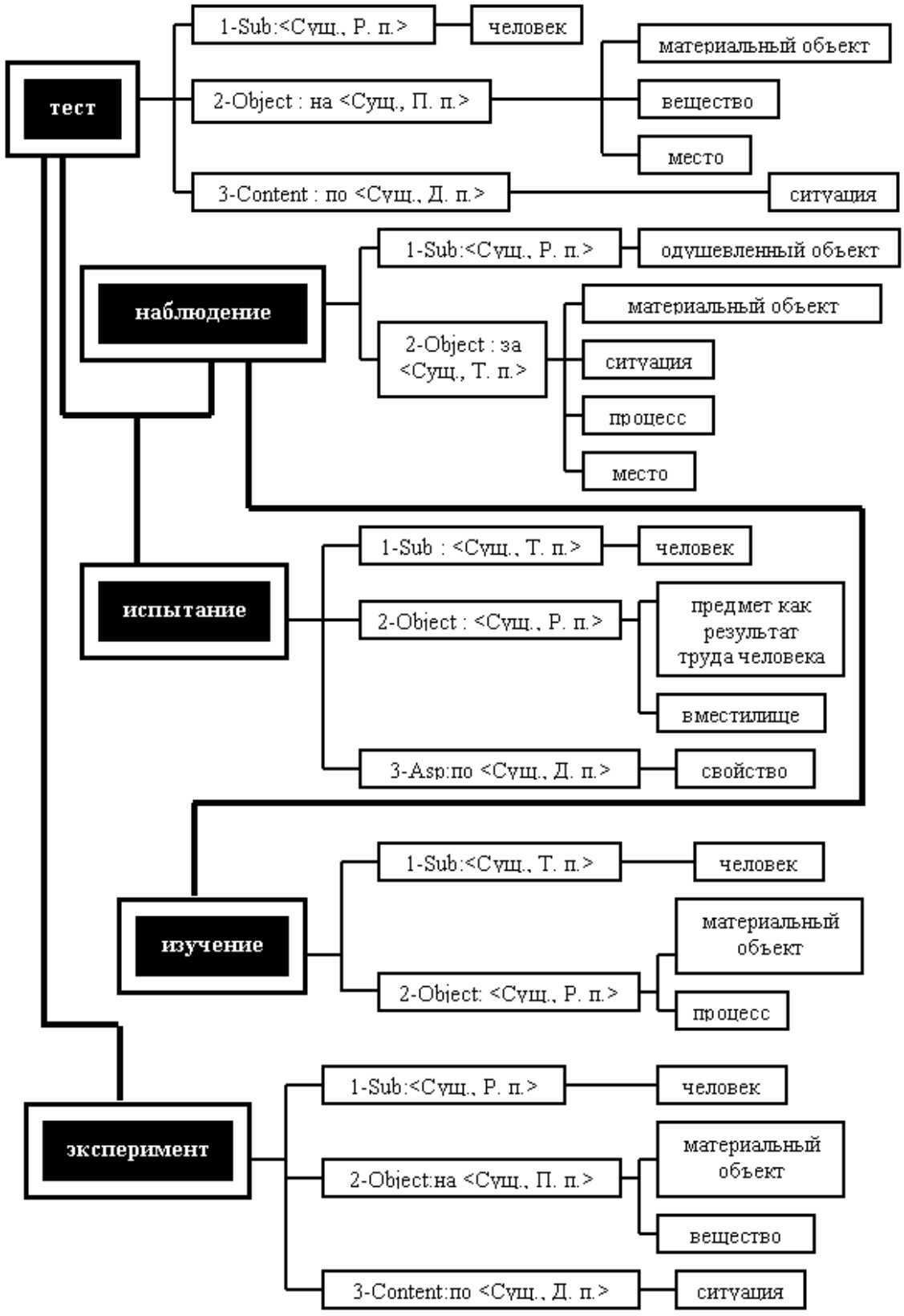


Рис. 3.9. Ролевой состав слов окрестности ЛЗ "*эксперимент*"

Таблица 3.1

Слова окрестности ЛЗ "*эксперимент*" и их семантические классы

Слово	Семантический класс
эксперимент	получение знаний об объекте или явлении при контролируемых условиях
испытание	действие с целью получения знаний при сопутствующем наблюдении
изучение	получение знаний
тест	действие с целью получения знаний
наблюдение	целенаправленное восприятие

Замечание. Фактически *Утверждением 3.6* определяется отношение порядка на множестве предикатных слов для случая зависимости между их семантическими характеристиками. При этом взаимно-однозначное соответствие между семантическими классами актанта гипонима и гиперонима устанавливается путем поиска общих подсписков семантических характеристик в совокупности с вхождением семантических характеристик одного актанта в утверждения теорий для семантических характеристик другого актанта.

Пусть W_1 и W_2 – комплексы лексических единиц, заменяемых посредством некоторого правила $\pi \in \Pi^R$ согласно постановке *Задачи 1.2*, W_1 отождествляется с РЗ, а W_2 – нерасщепленным смысловым эквивалентом этого РЗ. Положим также, что заданы структуры $Lm(w_1)$ и $Lm(w_2)$ вида (3.1) для ЛЗ слов $w_1 \in W_1$ и $w_2 \in W_2$, соответственно. Обозначим множество, каждый элемент которого входит либо в W_1, либо в W_2 и является предикатным словом, как W^S. При этом для каждого $w_i \in W^S$ имеется описание описание характеризованного ролевого состава посредством структуры (3.8).

Утверждение 3.7. Будем считать, что $Lm(w_1)$ и $Lm(w_2)$, $\{w_1, w_2\} \subset W^S$, адекватно задают $r(\pi)$ при выполнении следующих условий:

1. На множестве W^S может быть определено отношение порядка ($\leq$) в соответствии с условиями в *Утверждениях 3.5* и *3.6*.
2. Между w_2 и w_1 существует смысловое отношение F в соответствии с условиями, задаваемыми *Утверждением 3.4*.
3. Само имя отношения F в составе формального контекста (3.6) принадлежит множеству формальных признаков ЛЗ слова w_{Sup}, составляющего объем формального понятия, не превышающего наименьшего общего суперпонятия для множества N^H формальных понятий, объемы которых включают слова верхней окрестности ЛЗ w_1. Формально $N^H \subset \Re(G^H, M^H, V^H, I^H)$, при этом $G^H \supset W^S$, а M^H есть множество возможных ролевых ориентаций актантов (3.9) для обозначаемых предикатными словами $w_m \in G^H$ ситуаций. Множество V^H есть множество всех множеств семантических классов слов, способных замещать некоторую валентность R_{ti} предикатного слова $w_m \in G^H$, а $I^H \subseteq G^H \times M^H \times V^H$.

Требования к РЗ, в состав которого входит слово w_{Sup}, определяются аналогично.

3.4. Экспериментальная апробация методики формирования прецедентов смысловой эквивалентности на материале тезауруса по анализу изображений

Разработанная методика формирования прецедентов для классов СЭ, определяемых на основе расщепленных значений с лексическими функциями-параметрами, была апробирована на материале

специализированного тезауруса по анализу изображений, предложенного и развиваемого исследовательским коллективом Вычислительного центра им. А.А. Дородницына Российской академии наук. Концепции такого тезауруса и ее техническому воплощению был посвящен ряд публикаций наших коллег, в частности, [94,95,96,97,98,116].

Следует отметить, что формализация знаний в области обработки, анализа и понимания изображений является неотъемлемой составляющей построения интеллектуальных систем, способных выполнять функцию партнера человека при обработке больших массивов разнотипной информации, поступающей независимо из различных источников. Первым шагом на пути к созданию таких систем является построения онтологии той предметной области, которая включает обработку, анализ и распознавание изображений. При этом логико-понятийную основу онтологии составляет тезаурус, основным требованием к которому является динамичность. Тезаурус интеллектуальной системы должен быть не только средством представления современного состояния рассматриваемой области знания, включать все основные понятия и фиксировать существующие связи между этими понятиями, но и гибким инструментом интеграции новых знаний и уже имеющихся, обобщения и систематизации знаний, отслеживания противоречий в той информации, которая заносится в тезаурус.

Приведенный далее на рис. 3.10–3.17 пример показывает, каким образом предложенный в настоящей главе подход к описанию смысла слова набором характеристических функций позволяет решить указанные задачи, возлагаемые на тезаурус, а также уменьшить объем памяти ЭВМ, занимаемый самим тезаурусом.

Рис. 3.10. Вариант 1 теории ЛЗ "*изображение*"

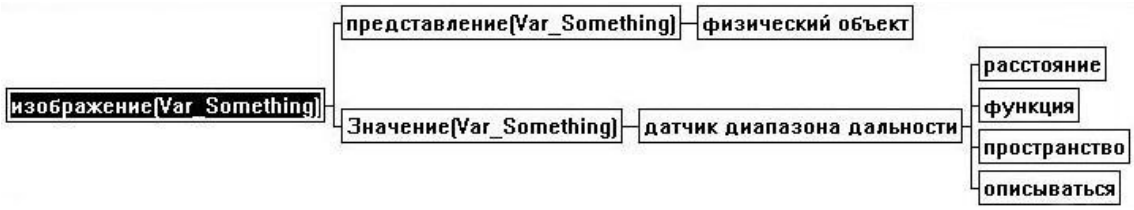


Рис. 3.11. Характеристические функции и формальные признаки их значений – вариант 1

Рис. 3.12. Вариант 2 теории ЛЗ "*изображение*"

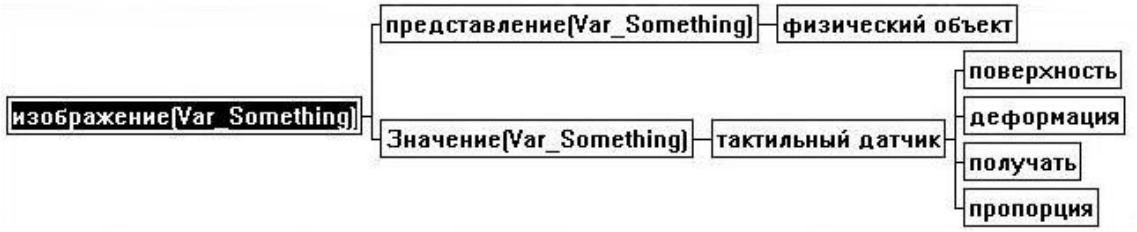


Рис. 3.13. Характеристические функции и формальные признаки их значений – вариант 2

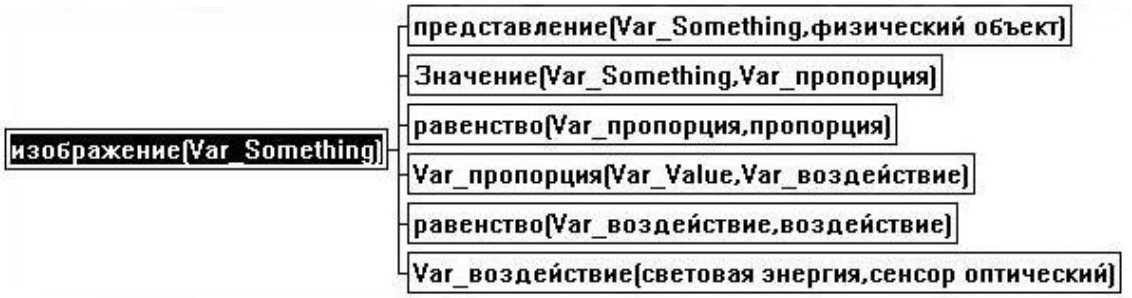


Рис. 3.14. Вариант 3 теории ЛЗ "*изображение*"

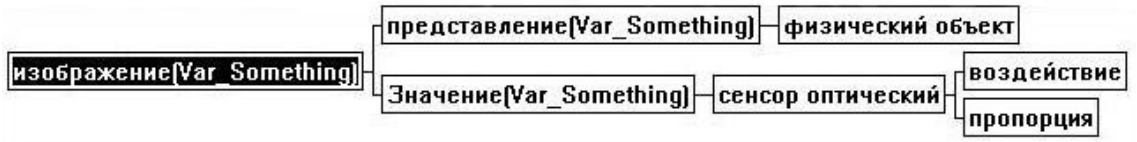


Рис. 3.15. Характеристические функции и формальные признаки их значений - вариант 3

При этом для обобщения независимых вариантов толкования лексического значения слова используются математические методы АФП и реализующее эти методы программное обеспечение, хорошо зарекомендовавшие себя в лингвистических приложениях [125] и свободно распространяемое в сети Internet. Это дает возможность распараллелить работу по созданию тезауруса заданной предметной области между исследовательскими коллективами разных научных школ, а посредством концептуальной кластеризации сопоставлять различные точки зрения на тот или иной термин (понятие).

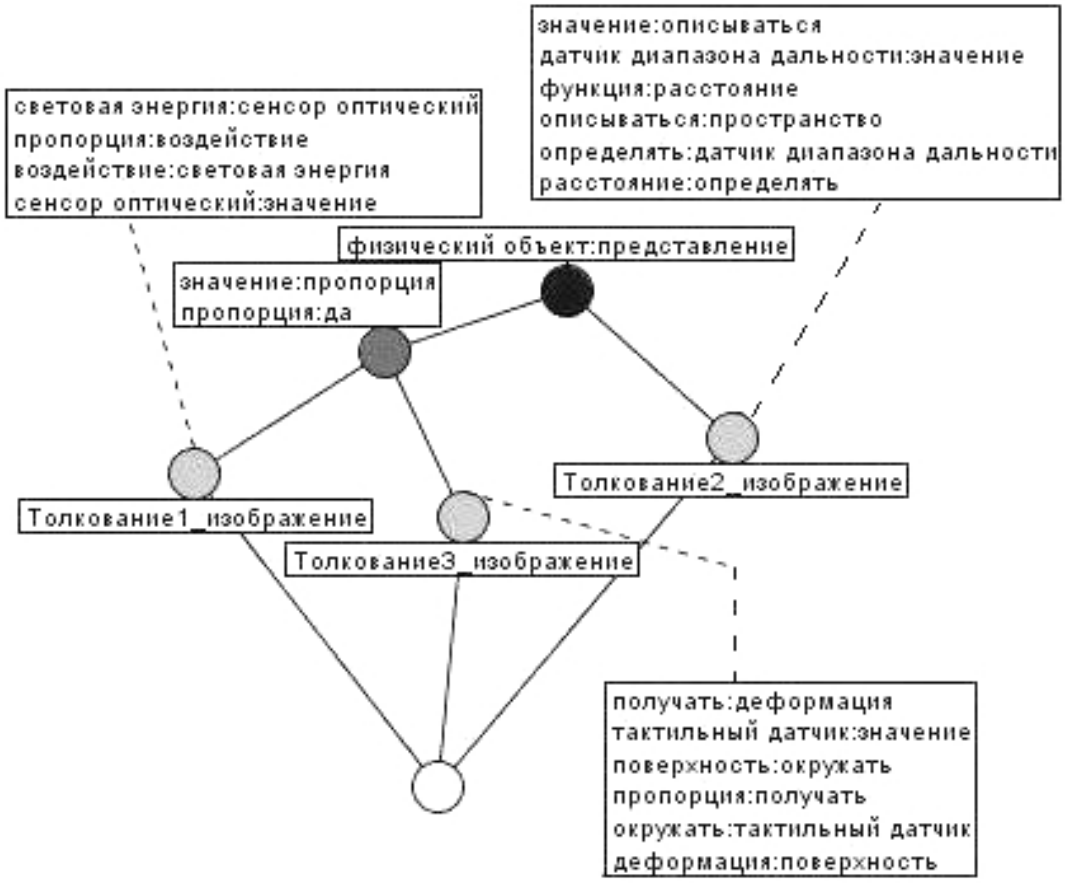


Рис. 3.16. Решетка формальных понятий для независимых толкований ЛЗ "*изображение*"

Задействование характеристических функций при описании смысла слова и их выводимость из теории его лексического значения позволяет в перспективе ввести в рассмотрение родовидовые зависимости между теориями на основе решеток, получаемых по нескольким независимым вариантам толкования одного и того же лексического значения (рис. 3.16). При этом базис импликаций [115] формального контекста (3.5) может послужить основой изучения взаимозаменяемости элементов толкования относительно различных характеристических функций.

Тем не менее, следует отметить, что основой информационного наполнения рассматриваемого тезауруса являются тематические публикации по заданной предметной области.

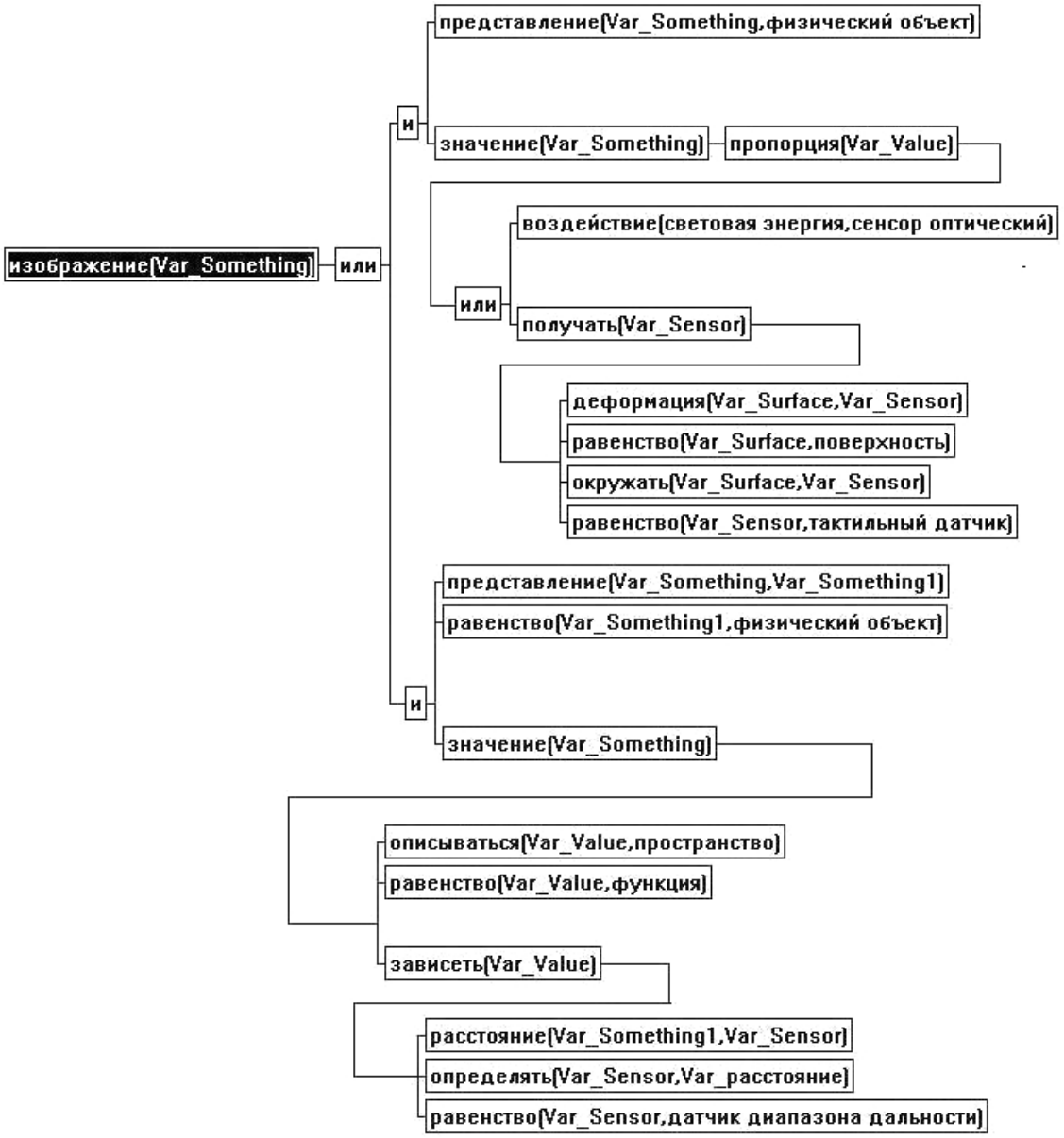


Рис. 3.17. Обобщение утверждений независимых теорий для ЛЗ "*изображение*"

На практике сказанное означает не только необходимость систематизации уже накопленных знаний, но и автоматизированное получение новых непосредственно из текстов (научных статей, тезисов докладов, монографий), формируемых носителем предметных знаний – человеком. В частности, для генерации структур вида (3.1) требуется решение задачи формирования и кластеризации отношений, на основе которых строятся утверждения теорий. Этому вопросу посвящен следующий раздел.

3.5. Формирование отношений в естественном языке на основе множеств семантически эквивалентных фраз

Как было показано нами в *Главе 1*, языковой опыт человека можно разделить в соответствии с разделением концептуальной картины мира. При этом основополагающим является понятие ситуации употребления ЕЯ как основы его генезиса, представляемой моделью вида (1.1). Предположим теперь, что в качестве элементов множества T в составе структуры (1.1) выступают синонимичные (с точки зрения носителя языка) ЕЯ-фразы, причем каждая из них описывает одну ситуацию действительности (относительно языкового контекста ситуации S). Положим выбор ЕЯ-фраз $T_i \in T$ для описания S равновероятным.

Поскольку S есть (по определению) полное и независимое описание языкового контекста, то имеем задачу:

Задача 3.1. На основе ЕЯ-фраз множества T сформировать отношения, представляемые множеством R в модели (1.1), рассматривая отношения между объектами $o \in O$ в качестве признаков последних относительно ситуации S.

Рассмотрим текст $T_i \in T$ с точки зрения символов, которые его составляют. Для $\forall T_i \in T$ справедливо:

$$T_i = T_i^C \cup T_i^F ,$$

где T_i^C – общая неизменная часть для всех $T_i \in T$, T_i^F – флективная часть.

На множестве T_i^F выражаются синтагматические зависимости, которые задаются с помощью R. Если $T_i = \bigcup_j W_{ij}$, то, соответственно,

$$W_{ij} = W_{ij}^{C} \cup W_{ij}^{F} . \qquad (3.10)$$

Здесь W_{ij} – буквенный состав слова, $W_{ij}^{C} \subset T_i^{C}$ – неизменная, $W_{ij}^{F} \subset T_i^{F}$ – флективная часть.

Таким образом, попарным сравнением W_{ij} различных T_i требуется найти:

1) W_{ij}^{C} и W_{ij}^{F} каждого W_{ij} при $\left|W_{ij}^{C}\right| \to \max$;

2) отношение R_q, определяющее допустимость сочетания $\left(W_{ij}^{F}, W_{ik}^{F}\right)$, $k \neq j$.

Введем в рассмотрение индексное множество J для неизменных частей всех слов, употребленных во всех фразах из T.

Определение 3.2. Моделью L линейной структуры предложения $T_i \in T$ будем называть упорядоченную совокупность индексов $j \in J$ неизменных частей слов, присутствующих в T_i.

При этом порядок индексов в L идентичен порядку следования соответствующих слов в T_i. Поэтому $L(T_i)$ позволяет однозначно восстановить ЕЯ-фразу T_i на множестве всех слов для всех фраз из множества T. И наоборот, для $\forall T_i \in T$ на индексном множестве J можно однозначно построить $L(T_i)$.

Для построения множества R в составе структуры (1.1) необходимо найти совокупность указанных моделей, удовлетворяющих требованиям проективности. С учетом линейной природы синтагм дополним ограничения на проективность [31], используемые в системах анализа текстов, следующим образом.

Пусть $h(j, L(T_i))$ – позиция индекса j в модели $L(T_i)$. Тогда множество связей относительно $L(T_i)$ можно определить как

$$D: T_i \to \{ (h(j, L(T_i)), h(k, L(T_i))) : j \neq k \}.$$

Определение 3.3. Связь $d_{qi} = (h(j, L(T_i)), h(k, L(T_i)))$ является *допустимой для модели* $L(T_i)$, если $\exists \{T_l, T_m\} \subset T$, $l \neq m$, причем и $L(T_l)$, и $L(T_m)$ содержат в качестве подпоследовательности либо $\{j,k\}$, либо $\{k, j\}$. При этом пара индексов (j,k) соответствует одной синтагме, а индекс q – типу синтаксического отношения, которое ей соответствует.

Положим, что для $\forall T_i \in T$, $i = 1,\ldots,|T|$, все $d_{qi} \in D(T_i)$ удовлетворяют *Определению 3.3*.

Определение 3.4. Будем считать, что модель $L(T_i)$ проективна относительно множества R в структуре (1.1), если $\sum_{q=1}^{|D(T_i)|} \Delta_{qi} \leq |L(T_i)|$, где $\Delta_{qi} = |h(j, L(T_i)) - h(k, L(T_i))|$.

На основе $\bigcup_i D(T_i)$ формируется граф синтагм (V^J, I^J). Элементами множества вершин V^J этого графа являются множества пар (j,k), $\{j,k\} \subset J$, сгруппированных по некоторому общему для них индексу k. Множества E_1 и E_2, входящие в V^J, будут соединены ребром из I^J, если $\exists \{j,k,m\} \subset J$: $(j,k) \in E_1$, $(k,m) \in E_2$ и $j \neq m$.

Анализом (V^J, I^J) строится дерево-прецедент (V_1^J, I_1^J) для $\bigcup_i T_i$, $i = 1,\ldots,|T|$. Формально

$$V_1^J = J,\ I_1^J = \{(j,k) : \exists E \in V^J, (j,k) \in E\}. \qquad (3.11)$$

При этом индекс $k \in V_1^J$ соответствует корню дерева (V_1^J, I_1^J), если $\exists E_1 \in V^J$, в котором пары индексов сгруппированы по k, $|E_1| > 1$, а k не содержится ни в одной паре индексов для $\forall E_2 \in V^J$: $E_1 \neq E_2$.

Содержательно корень соответствует предикатному слову (глаголу, либо отглагольному существительному), которое (по определению) обозначает ситуацию. Согласно данному в *Главе 1* определению семантического отношения, наибольший интерес для *Задачи 3.1* представляют ситуации вида (1.1) с двумя и более участниками, поэтому число дочерних узлов у корня полагается больше одного.

Будем использовать маршруты в дереве (3.11) для выделения классов отношений множества R в модели (1.1) согласно сформулированной нами *Задаче 3.1*. Данная задача наиболее естественно решается методами АФП.

Рассмотрим множество флексий как множество формальных объектов $G^F = \left\{ f_{ij} : f_{ij} = \bullet\left(W_{ij}^F\right) \right\}$, где $i = 1,\ldots,|T|$, а символом "$\bullet$" обозначается операция конкатенации, которая последовательно выполняется над символами из W_{ij}^F.

Введем в рассмотрение формальный контекст:

$$K^F = \left(G^F, M^F, I^F\right), \tag{3.12}$$

в котором $M^F = G^F$, а $I^F \subseteq G^F \times M^F$. При этом

$$I^F = \left\{ \left(f_{ij}, f_{ik}\right) : s(j,k) = true, \{j,k\} \subset J \right\}.$$

Отношение s определяется рекурсивно на основе $\left(V^J, I^J\right)$:

1) $s(j_1, j_1) = true$;

2) $s(j_1, j_2) = true$ в одном из следующих двух случаев:

– $\exists E_1 \in V^J$: $(j_1, j_2) \in E_1$, причем $\exists j_3 \in J$, для которого $s(j_2, j_3) = true$;

– $\exists (E_1, E_2) \in I^J$: $\exists j_3 \in J$, при этом $(j_1, j_3) \in E_1$, $(j_3, j_2) \in E_2$, а $s(j_3, j_2) = true$.

Модель (3.12) выделяет классы в R по характеру изменения флективной части зависимого слова в каждом из отношений $R_q \in R$ с учетом бинарности последнего.

Рассмотрим задачу поиска флексий для слов в составе расщепленных значений, семантику которых мы обсуждали в *Разделе 3.3*. Здесь мы рассмотрим общий случай Расщепленного Предикатного Значения (РПЗ) как совокупности вспомогательного глагола (связки) и некоторого существительного, называющего ситуацию. Для слов в составе РПЗ, как и для конверсивов (слов, обозначающих ситуацию с точки зрения разных ее участников) представления вида (3.10) не могут быть найдены попарным сравнением буквенного состава слов во всех $T_i \in T$.

Рассмотрим $T_i^{Cnc} = \{w_{ij} : w_{ij} = \bullet(W_{ij})\}$. Положим также, что $\exists T_i^P \subset T_i$, определяющее последовательность:

$$P_i^{Cnc} = \left\{u_k : u_k = \bullet\left(W_k^P\right), \bigcup_k W_k^P = T_i^P\right\},$$

где $W_k^P \in T_i$ – последовательность символов слова, для которого не найдено представления (3.10).

Лемма 3.1. Последовательность P_i^{Cnc} содержит предикатное слово, если $\exists\{j,0,k\} \subset L(T_i)$: $\{w_{ij}, u_1, \ldots, u_p, w_{ik}\} \subset T_i^{Cnc}$, где $\{u_1, \ldots, u_p\} = P_i^{Cnc}$, $p = \left|P_i^{Cnc}\right|$.

Доказательство следует из определения корня дерева $\left(V_1^J, I_1^J\right)$ и сделанного допущения о числе участников ситуации (1.1) с учетом проективности $L(T_i)$.

Пусть для последовательности P_i^{Cnc} выполняется условие *Леммы 3.1*.

Лемма 3.2. Слово $u_k \in P_i^{Cnc}$ принадлежит РПЗ, если $\exists T_j \in T$: $L(T_j) \neq L(T_i)$, а $u_k \in P_j^{Cnc}$, где P_j^{Cnc} также отвечает условию *Леммы 3.1*. При этом $\neg\exists T_k \in T$: $P_k^{Cnc} \subset P_i^{Cnc}$, а $L(T_k) \neq L(T_j)$ и $L(T_k) \neq L(T_i)$.

Доказательство следует из доказанной *Леммы 3.1* и определения множества ребер в графе (V^J, I^J).

Замечание. При выполнении условия *Леммы 3.2* u_k может быть в том числе и зависимым словом в составе РПЗ.

Пусть $P_i^{Cnc'}$ – последовательность слов, удовлетворяющих условию *Леммы 3.2*.

Теорема 3.1. Для формирования структуры (3.12) при наличии РПЗ либо конверсива необходимо и достаточно найти множество $T' \subset T$:

$$T' = \left\{ T_i : \left| P_i^{Cnc'} \right| \to \max \right\}.$$

Доказательство следует из доказанной *Леммы 3.2*.

Помимо выполнения условия *Теоремы 3.1*, ключевым требованием при отборе $T_i \in T$ является минимум слов, не представимых соотношением (3.10). Для $\forall u_k \in \bigcup_i P_i^{Cnc'}$, $T_i \in T'$, представление вида (3.10) формируется сравнением буквенного состава со всеми $u_j \in \bigcup_l P_l^{Cnc}$: $T_l \in (T \setminus T')$. При этом необходимо, чтобы $2\left|W_k^C\right| > \left|W_k^F\right| + \left|W_j^F\right|$, где $W_k^P = W_k^C \cup W_k^F$, а $W_j^P = W_j^C \cup W_j^F$.

Замечание. Если $P_i^{Cnc'} \cap P_i^{Cnc} \neq \varnothing$, то $\forall u_m \in \left(P_i^{Cnc} \setminus P_i^{Cnc'}\right)$ есть предлог и представляется вместе со словом, стоящим слева от него в последовательности P_i^{Cnc}.

С учетом $P_i^{Cnc'}$ дерево (3.11) преобразуется следующим образом:

1) корень изменяется с $k=0$ на значение k для $u_k \in P_i^{Cnc'}$, имеющего максимальную встречаемость в различных T_i^{Cnc} относительно заданной ситуации языкового употребления;
2) левое поддерево остается без изменений;
3) правое поддерево перевешивается на узел j для $u_j \in P_i^{Cnc'}$ наименьшей встречаемости;
4) в паре $\{u_l, u_m\} \subset P_i^{Cnc'}$ дочерним будет узел для слова с меньшей встречаемостью.

В итоге основу формирования модели (3.12) составляют те T_i, которые наиболее полно представляют языковой контекст заданной ситуации (1.1).

В заключении данного раздела рассмотрим свойства формального контекста (3.12), актуальные для выделения морфологических классов слов из множества T', сформированного в соответствии с *Теоремой 3.1*.

Пусть ℓ – базис импликаций, а $\Re^F$ – решетка формальных понятий для формального контекста K^F.

Утверждение 3.8. ФП $\left(A^F, B^F\right): A^F \subseteq G^F, B^F \subseteq M^F$ соответствует предикатному слову, если $\exists(\mathrm{Pr} \to Cs) \in \ell:\ |\mathrm{Pr}| = 1$ и $\mathrm{Pr} \cup Cs = B^F$. При этом наличие импликации $(\mathrm{Pr}_1 \to Cs_1) \in \ell : \mathrm{Pr} \subset Cs_1$ допускается только тогда, когда $\mathrm{Pr}_1 \cup Cs_1 = B^F$.

Утверждение 3.9. ФП $\left(A^F, B^F\right): A^F \subseteq G^F, B^F \subseteq M^F$ соответствует слову, выполняющему в ЕЯ-фразе функцию

определения (прилагательному либо причастию не в составе оборота), если B^F есть множество признаков некоторого элемента множества G^F и $\neg\exists(\Pr \to Cs) \in \ell: \Pr \cup Cs = B^F$. Элементами B^F при этом должны быть непустые строки. Если же множество B^F состоит из единственного элемента – пустой строки, то данное ФП соответствует слову с синтаксической функцией наречия.

В противном случае ФП $\left(A^F, B^F\right)$ соответствует слову, выполняющему синтаксическую функцию существительного.

Отношения, представляемые множеством R в модели (1.1), выделяются анализом наименьшей верхней грани каждой пары ФП в $\mathfrak{R}^F$ и образуют классы по сходству характера флексии зависимого слова. Отдельному классу соответствует область в решетке, а наименьшая верхняя грань множества формальных понятий этой области – прецеденту класса. Следует отметить, что в настоящем разделе мы ведем рассмотрение только синтагматических зависимостей. Более широкие классы отношений, определяемые сочетанием основ главного и зависимого слова, а также сочетанием основ и флексий, выделяются аналогично. О формировании этих отношений пойдет речь в следующей главе работы.

В качестве примера рассмотрим выделение и классификацию синтаксических отношений на множестве вариантов правильного ответа для тестового задания открытой формы.

Вопрос теста: “Каковы негативные последствия переобучения при скользящем контроле?” В итоге было получено двадцать семь вариантов правильного ответа на данный вопрос (рис. 3.18).

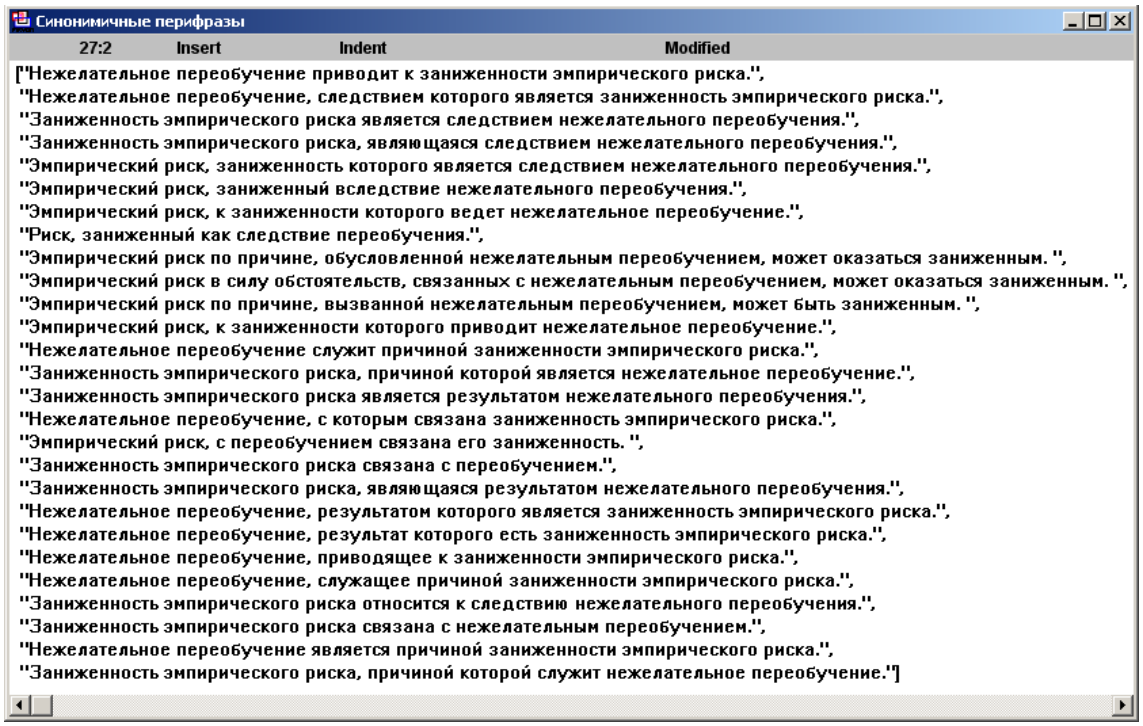


Рис. 3.18. Исходные данные для формирования модели (3.12)

Таблица 3.2

Правильные ответы $T_i \in T'$

Основа	Флективная часть + предлог					
заниженн	ость	ости	ость	ости	ость	ости
эмпирическ	ого	ого	ого	ого	ого	ого
риск	а	а	а	а	а	а
нежелательн	ого	ое	ого	ое	ым	ое
переобучени	я	е	я	е	ем	е
явля	ется	–	ется	ется	–	–
следстви	ем	–	–	–	–	–
служ	–	ит	–	–	–	–
причин	–	ой	–	ой	–	–
результат	–	–	ом	–	–	–
связан	–	–	–	–	а:с	–
привод	–	–	–	–	–	ит:к

При этом основу формирования решетки $\mathfrak{R}^F$, представленной на рис. 3.19, составили максимально проективные ЕЯ-фразы с минимумом слов, не нашедших прообразов по буквенному составу.

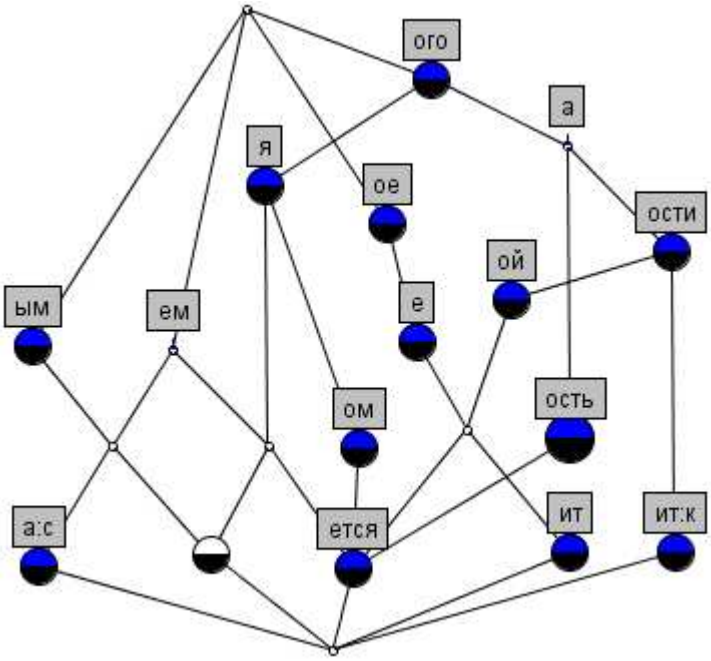


Рис. 3.19. Синтаксические отношения на основе сочетаний флексий

Визуализацию решетки диаграммой линий здесь и далее выполняет программная система “Concept Explorer” [126], реализующая методы АФП.

Содержательная интерпретация решетки $\mathfrak{R}^F$ может быть получена выделением морфологических классов слов на основе базиса импликаций, представленного на рис. 3.20.

В приведенном на рис. 3.19 примере классы отношений соответствуют словоизменению прилагательных (*нежелательн-ого*, *эмпирическ-ого*) и существительных в составе генитивных конструкций (*результат-ом переобучени-я*, *следстви-ем переобучении-я*). Последний в силу транзитивности синтаксического отношения в рамках последовательности соподчиненных слов может включать сочетания существительного (вне генитивных конструкций) с глаголом. Более подробно это отношение будет рассмотрено в следующей главе работы.

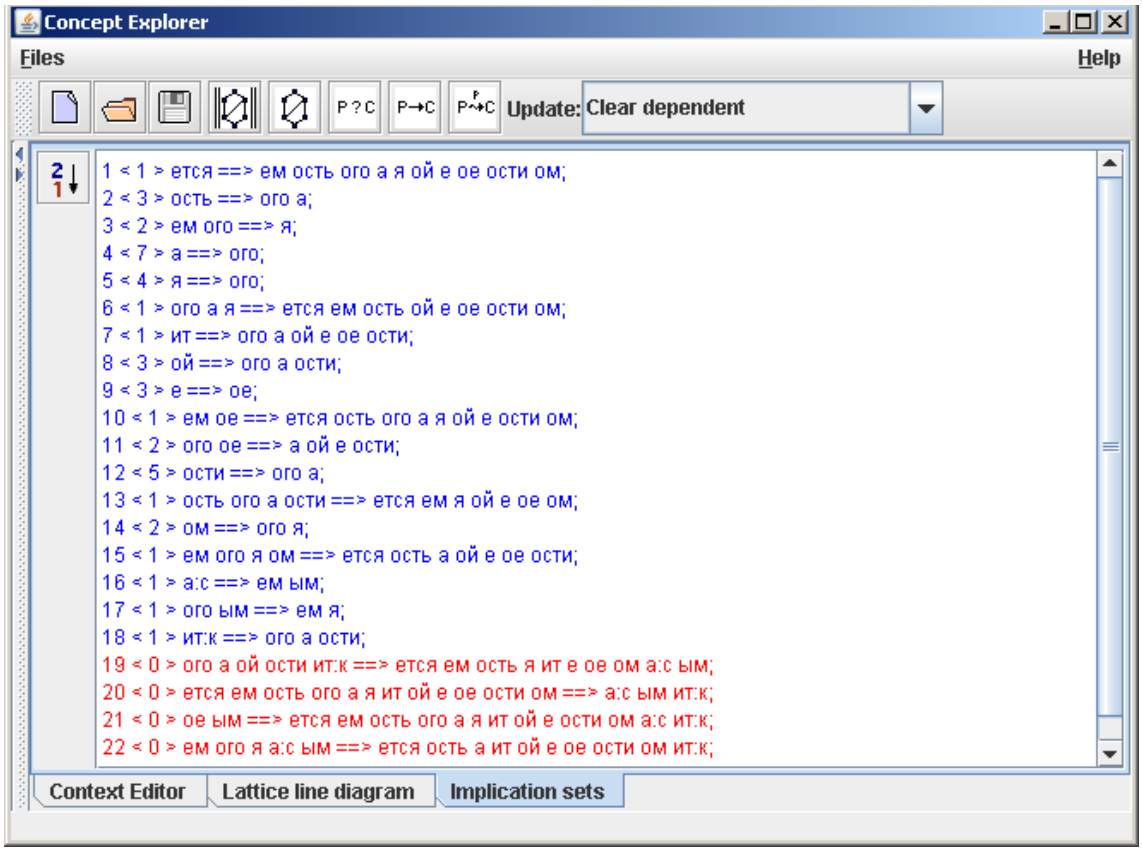


Рис. 3.20. Базис импликаций на основе результирующего множества ЕЯ-фраз

Поскольку основу формирования решетки $\mathfrak{R}^F$ составляют те ЕЯ-фразы, которые максимально точно описывают ситуацию, а значит и более четко передают смысл согласно данному в *Разделе 3.1* настоящей главе формальному определению смысла, то выявленные отношения будут соответствовать искомым наиболее вероятным синтаксическим связям относительно модели (1.1).

Выводы

Предложенный в главе подход к выделению и классификации синтагматических зависимостей позволяет выделять любые отношения в тексте, в том числе за рамками синтаксиса простого распространенного предложения.

При описании семантических отношений в предикатной форме [33] теоретико-решеточное представление связи между различными аргументами отношения позволяет просто и естественно показать выражение предиката семантического отношения через комбинацию более простых тем самым наглядно проиллюстрировать понятие сложности предиката.

Введение характеристических функций для элементов толкований лексических значений слов позволяет наряду с описанием условий применимости для правил синонимических преобразований на уровне глубинного синтаксиса, на основе формального контекста элементов толкования формализовать процедуру анализа сходства самих правил, а также устанавливать близость наборов таких правил, о которой говорилось в [19].

Отметим, что предложенное в настоящей главе описание смысла слова набором характеристических функций производится в шкале наименований. При обобщении утверждений независимых теорий одного и того же лексического значения посредством отношения "или" не учитывается статистическая значимость каждого признака. Значения характеристических функций, задаваемые объединяемыми утверждениями, полагаются равновероятными.

Для введения в рассмотрение, к примеру, распределений возможных значений характеристических функций необходимо учитывать семантические свойства синтаксического контекста слова (в первую очередь – контекста существительного), который служит (по определению) базой формирования отношений в рамках формализованной теории лексического значения. Семантике синтаксического контекста имени существительного как основы кластеризации текстов посвящается следующая глава работы.

Глава 4

СЕМАНТИЧЕСКАЯ КЛАСТЕРИЗАЦИЯ ТЕКСТОВ ЕСТЕСТВЕННОГО ЯЗЫКА НА ОСНОВЕ СИНТАКСИЧЕСКИХ КОНТЕКСТОВ СУЩЕСТВИТЕЛЬНЫХ

Основная задача, решаемая в данной главе – использование синтаксических отношений в текстах как основы их кластеризации. При этом основной акцент внимания уделяется семантическим аспектам синтаксиса как такового. На основе свойств соотношения смыслов соподчиненных слов решается задача установления частичных СЭ. Рассматривается использование синтаксического контекста имени существительного как основы выделения объектов и ситуаций, описываемых сравниваемыми текстами. Рассматривается критерий полезности решетки формальных понятий и его использование для определения силы семантической связи слов и в качестве основы систематизации конверсивов и расщепленных предикатных значений в рамках рассматриваемого синтаксического контекста.

4.1. Семантика синтаксиса как основа кластеризации

Как было показано нами в предыдущей главе, лексическая сочетаемость слова зависит от его семантического класса. Поэтому справедливо предположение о возможности выявления СК слова анализом его сочетаний с другими словами в ЕЯ-текстах по тематике заданной предметной области.

Следует отметить, что первостепенную роль для извлечения СК слова из набора текстов заданной тематики играет контекст целевого слова.

Наибольшую точность, как показывает практика, дают модели контекста на основе синтаксических связей в предложении [82, 124].

В двух предыдущих главах основной акцент нашего внимания был уделен контексту предикатного слова, который определяется, в первую очередь, синтаксическими связями между предикатом и его семантическими актантами. Согласно постановке *Задачи 1.1*, для формализации понятий Предметной Области, обозначающих участников тех или иных ситуаций, необходимо ввести в рассмотрение сочетаемость соответствующих существительных со словами, являющимися синтаксически главными по отношению к ним. Причем наряду с сочетаниями "актант-предикат" требуется учитывать произвольные сочетания существительных в тексте между собой (в том числе посредством предлогов).

Каждое выявляемое из текста понятие идентифицируется (в первую очередь) относительно заданного множества ситуаций.

Поскольку сами ситуации обозначаются предикатными словами – глаголами либо их производными, наиболее приемлемым вариантом синтаксического контекста для существительного, обозначающего некоторое выявляемое понятие, будет последовательность соподчиненных слов:

$$S_{ki} = \{v_1, \ldots, v_{n(k,i)}, m_{ki}\}, \qquad (4.1)$$

где v_1 – предикатное слово, которое обозначает ситуацию;

m_{ki} – существительное и обозначает некоторое понятие, значимое в ситуации v_1 из описываемых текстом T_i;

$\forall v_l \in \{v_2, \ldots, v_{n(k,i)}\}$ – некоторое существительное;

k – порядковый номер последовательности среди выявленных из текста T_i;

$n(k,i)$ – количество соподчиненных существительных последовательности.

При использовании последовательности (4.1) как основы выделения элементов множества O в составе структуры вида (1.1) множество R такой структуры составят синтаксические отношения R_q:

$$v_l R_q v_{l+1}, \ldots, v_{n(k,i)} R_q m_{ki} \tag{4.2}$$

для всех S_{ki}, $i = 1,\ldots,|T|$. Здесь индекс q соответствует типу отношения R_q, который характеризуется падежом зависимого слова и предлогом для связи главного и зависимого слова. При этом q соответствует имени синтагмы, которая определяет бинарное отношение вида (4.2).

Введение в рассмотрение синтаксического контекста вида (4.1) дает основание предположить возможность наличия для любого текста T_i множества T в составе структуры (1.1) последовательности $S_{lki} \neq S_{ki}$:

$$S_{lki} = \{v_l, m_{ki}\} \tag{4.3}$$

для $\forall v_l \in \{v_1,\ldots,v_{n(k,i)-1}\}$, где $v_l R_q m_{ki}$. При этом обязательным является наличие $v_l R_q v_{l+1}$ в рамках последовательности (4.1). Будем называть последовательность S_{ki} ситуационным контекстом для m_{ki}. В этом случае S_{ki} в совокупности с множеством $\{S_{lki}\}_{l=1}^{n(k,i)-1}$ определяют некоторые ситуации (либо ассоциируемые с ними понятия) относительно m_{ki}. Причем с любой S_{lki} связывается более абстрактное понятие (ситуация), чем с S_{ki}.

Утверждение 4.1. При одновременном наличии последовательностей $S_{ki} = \{v_1, \ldots, v_{n(k,i)}, m_{ki}\}$ и $S_{1ki} = \{v_1, m_{ki}\}$ в разных текстах множества T имеет место частичная СЭ (относительно m_{ki}).

Пример: "Характеристика сложности семейства алгоритмов"⇔"характеристика алгоритмов". Подобная СЭ может задаваться, в частности, генитивной конструкцией [82, 124]. Для сравнения: "сложность подсемейства модели"⇔"сложность модели".

Утверждение 4.2. При наличии отношения R_q между v_1 и v_2 возможно установление указанного отношения между v_1 и любым словом последовательности (4.1) вне зависимости от существующих отношений.

Доказательство следует из соотношения смыслов соподчиненных слов. При этом для установления отношения R_q между v_1 и произвольным v_l, $l = 3, \ldots, n(k,i)$, а также между v_1 и m_{ki} зависимое слово должно быть приведено в соответствующую морфологическую форму.

Пример. Рассмотрим словосочетание *"рассматривать на множестве семейств алгоритмов"*. Допустимыми с точки зрения синтаксиса и семантики русского языка являются также словосочетания *"рассматривать на семействах"* и *"рассматривать на алгоритмах"*.

В настоящей работе в качестве базовой структуры для выявления и кластеризации понятий мы будем использовать ситуационные контексты вида (4.1), которые участвуют в описании частичных СЭ в соответствии с *Утверждением 4.1*.

Ставится *задача*: путем синтаксического разбора предложений выявить указанные контексты в анализируемом тексте и на их основе выполнить концептуальную кластеризацию.

4.2. Концептуальная кластеризация текстов на основе результатов синтаксического разбора предложений

Результатом синтаксического анализа текста является набор деревьев разбора предложений. В настоящей работе синтаксический анализ осуществляется программой “Cognitive Dwarf” [75]. При тестировании данная программа показала самые точные результаты разбора.

На основе полученного набора деревьеа формируются ситуационные контексты (4.1). При этом с каждого дерева последовательно считываются пары (x,y), где x – синтаксически главное слово, y – зависимое слово. Дальнейшая обработка считанных пар направлена на выявление последовательностей (4.1) и (4.3) в соответствии с *Утверждением 4.1*. Обозначим множество последовательностей вида (4.1), формируемое относительно текста T_i, как P_i^S.

В качестве инструмента концептуальной кластеризации ситуационных контекстов (4.1) как основы выделения понятий будем использовать методы АФП, рассмотренные нами в предыдущих главах. Согласно постановке *Задачи 1.1*, имеем формальный контекст:

$$K=(G,M,V,I), \tag{4.4}$$

где $G \supset T$; V есть множество ситуаций, описываемых текстами из множества G; M есть множество объектов и/или понятий, значимых в ситуациях из множества V; $I \subseteq G \times M \times V$.

Замечание. На основе *Утверждения 4.2* справедливым будет утверждать, что $\forall v_l \in \{v_2,\ldots,v_{n(k,i)}\}$ в составе последовательности (4.1) обозначает некоторое понятие, значимое в ситуации v_1, наравне с

m_{ki}. Таким образом, если $V(T_i)$ есть множество ситуаций, описываемых текстом T_i, а $M(T_i)$ есть соответствующее ему множество объектов согласно постановке *Задачи 1.1*, то для любой S_{ki} $\{v_2,\ldots,v_{n(k,i)},m_{ki}\}\subset M(T_i)$. Причем $V(T_i)=\bigcup_k(S_{ki}\setminus\{m_{ki}\})$.

С учетом сказанного имеем расширение множеств $M(T_i)$ и $V(T_i)$ в соответствии с представленным ниже алгоритмом.

Алгоритм 4.1. Формирование троек-кандидатов на включение в отношение I.

Вход: P_i^S ; // множество последовательностей вида (4.1)

Выход: $P_i^K=\{P_{ki}^K : P_{ki}^K=\{(g_i,m,v):(g_i,m,v)\in I\}\}$;

// g_i есть некоторая пометка для $T_i\in G$

Начало

$P_i^K:=\varnothing$; // Инициализация

Начало цикла. Пока $P_i^S\neq\varnothing$

Выбрать S_{ki} из P_i^S;

$P_{ki}^K:=\varnothing$;

Начало цикла. Для $l=1,\ldots,n(k,i)$

$P_{ki}^K:=P_{ki}^K\cup\{(g_i,m_{ki},v_l)\}$;

// $S_{ki}=\{v_1,\ldots,v_{n(k,i)},m_{ki}\}$ в соответствии с (4.1)

$j:=n(k,i)$;

Начало цикла. Пока $j>l$

$P_{ki}^K:=P_{ki}^K\cup\{(g_i,v_j,v_l)\}$;

$j := j - 1$;

Конец цикла {Пока $j > l$ *}*;

Конец цикла {Для $l = 1, \ldots, n(k,i)$ *}*;

$P_i^K := P_i^K \cup \{P_{ki}^K\}$;

$P_i^S := P_i^S \setminus \{S_{ki}\}$;

Конец цикла {Пока $P_i^S \neq \varnothing$ *}*;

Конец {Алгоритм 4.1}.

При этом роль, в которой объект $m \in M(T_i)$ выступает относительно некоторой ситуации $v \in V(T_i)$, определяется типом q отношения R_q между словом v и словом справа от него в последовательности (4.1). Указанный тип характеризуется падежом зависимого слова и предлогом для связи синтаксически главного и зависимого слова. По этой причине каждое $v \in V(T_i)$ в составе троек, формируемых *Алгоритмом 4.1*, в зависимости от наличия/отсутствия предлога p_y между главным и зависимым словом представлено как:

$$v = \begin{cases} x \bullet ":" \bullet p_y \\ x \end{cases},$$

где x – синтаксически главное; y – зависимое слово; $\bullet$ – операция конкатенации. Для использования в дальнейших рассуждениях введем следующие функции: $prep : v \to p_y$, которая ставит в соответствие каждому $v \in V(T_i)$ предлог для связи с зависимым словом; $case : m \to c_y$, которая ставит в соответствие каждому именному $m \in M(T_i)$ символьное обозначение его падежа

$c_y \in \{"nom","gen","dat","acc","ins","loc"\}$. Соответствие между словом и его начальной формой зададим с помощью функции $norm$.

Основные этапы построения решетки ФП $\Re(G,M,V,I)$ для формального контекста (4.4) представлены *Алгоритмом 4.2*.

Алгоритм 4.2. Построение формального контекста для исходного множества текстов.

Вход: G ; // Исходное множество ЕЯ-текстов, $n(G)=|G|$

Выход: $K=(G,M,V,I)$; // Формальный контекст вида (4.4)

Начало

Шаг 1: Синтаксический анализ текстов из множества G с формированием множества P_i^S для каждого $T_i \in G$;

Шаг 2: Для $\forall T_i \in G$ на основе P_i^S выделить $M(T_i)$ и $V_1(T_i) \subset V(T_i)$: $V_1(T_i)=\{v_1 : \exists S_{ki} \in P_i^S, S_{ki}=\{v_1,\ldots,v_{n(k,i)}, m_{ki}\}\}$;

Шаг 3: На основе выделенных $\{M(T_i) \mid i=\overline{1,n(G)}\}$ и $\{V_1(T_i) \mid i=\overline{1,n(G)}\}$ найти одноименные ситуации v, принадлежащие различным $V_1(T_i)$ и сходные по фигурирующим в них объектам $m \in M$: $M=\bigcup_i M(T_i)$ в сходных ролях;

Шаг 4: Приписать названиям ситуаций, выделенных на *Шаге 3*, одинаковые индексы в соответствующих $V_1(T_i)$ и P_i^S;

Шаг 5: По аналогии с *Шагом 3* на основе P_i^S найти разноименные ситуации v, принадлежащие различным $V_1(T_i)$ и сходные по фигурирующим в них объектам $m \in M$ в сходных ролях;

Шаг 6: По каждой выявленной на *Шаге 5* группе синонимов $Syn = \{v_1 : S_{ki} = \{v_1, \dots, v_{n(k,i)}, m_{ki}\} \mid i = \overline{1, n(G)}\}$ выделить канонический представитель v_1' с наибольшей частотой употребления и заменить все $v_1 \in S_{ki} \ : S_{ki} \in Syn$, на v_1';

Шаг 7: Выполнить *Шаги 3-6* для разноименных ситуаций, принадлежащих различным $V_1(T_i)$ и сходных по фигурирующим в них $m \in M$, но с меной ролей (конверсивы);

Шаг 8: Для каждого текста $T_i \in G$ сформировать $V(T_i) = V_1(T_i) \cup \left(\bigcup_k (S_{ki} \setminus \{m_{ki}\} \setminus \{v_1\}) \right)$ и установить отношение I в соответствии с *Алгоритмом 4.1* с учетом результатов *Шагов 3-7*;

Конец {Алгоритм 4.2}.

Данный алгоритм описывает формирование множества ФП $\{(A, B) : A \subseteq G, B \subseteq M \times V, A = B', B = A'\}$ контекста (4.4). Здесь $V = \bigcup_i V(T_i)$, $M = \bigcup_i M(T_i)$ согласно введенным ранее обозначениям, A – объем, B – содержание формального понятия (A, B) согласно *Определению 1.10*, причем $A' = \{(m, v) : m \in M, v \in V \mid \forall g \in A : m(g) = v\}$, $B' = \{g \in G \mid \forall (m, v) \in B : m(g) = v\}$. При этом решетка $\Re(G, M, V, I)$ дает требуемую классификацию текстов исходного множества G относительно описываемых текстами ситуаций и фигурирующих в этих ситуациях объектов.

4.3. Расщепленные предикатные значения и конверсивы в составе синтаксических контекстов существительных

При формировании множеств объектов и ситуаций на основе синтаксического анализа исходных текстов актуальна проблема наличия расщепленных значений в составе последовательностей (4.1).

В настоящей главе за основу механизма выявления РЗ мы возьмем правила синонимических преобразований типа замещения с расщеплением в рамках стандартных ЛФ. Фактически именно на эти правила мы ссылались в предыдущей главе при формализации смыслового отношения в рамках расщепленного значения.

Пусть $\{T_1, T_2\} \subset G$ есть пара анализируемых текстов, $S_1 \subset T_1$, $S_2 \subset T_2$, $S_1 = \{S_{k1} | k = \overline{1, n(S_1)}\}$, где $n(S_1) = |S_1|$, а

$$S_2 = \begin{cases} \{S_{k2} | k = \overline{1, n(S_1)}\} \\ \{S_{k2} | k = \overline{1, n(S_1) - 1}\}. \end{cases}$$

Утверждение 4.3. Применительно к паре (T_1, T_2) расщепленное предикатное значение $\{v_{11}, v_{12}\}$ будет иметь место в следующих двух случаях.

Случай 1.

$$\left.\begin{array}{l} S_{11} = \{v_{11}, v_{12}, v_{13}, \ldots, v_{1,idx(1,1)}, m_{11}\} \\ S_{21} = \{v_{11}, v_{12}, v_{23}, \ldots, v_{2,idx(2,1)}, m_{21}\} \\ \ldots \\ S_{k1} = \{v_{11}, v_{12}, v_{k2}, \ldots, v_{k,idx(k,1)}, m_{k1}\} \\ \\ S_{k+1,1} = \{v_{11}, v_{k+1,2}, \ldots, v_{k+1,idx(k+1,1)}, m_{k+1,1}\} \\ \ldots \\ S_{n(S_1),1} = \{v_{11}, v_{n(S_1),2}, \ldots, v_{n(S_1),idx(n(S_1),1)}, m_{n(S_1),1}\} \end{array}\right\},$$

$$\left.\begin{array}{l} S_{12} = \{v_{21}, v_{13}, \ldots, v_{1,idx(1,1)}, m_{11}\} \\ S_{22} = \{v_{21}, v_{23}, \ldots, v_{2,idx(2,1)}, m_{21}\} \\ \ldots \\ S_{k2} = \{v_{21}, v_{k2}, \ldots, v_{k,idx(k,1)}, m_{k1}\} \\ S_{k+1,2} = \{v_{21}, v_{k+1,2}, \ldots, v_{k+1,idx(k+1,1)}, m_{k+1,1}\} \\ \ldots \\ S_{n(S_1),2} = \{v_{21}, v_{n(S_1),2}, \ldots, v_{n(S_1),idx(n(S_1),1)}, m_{n(S_1),1}\} \end{array}\right\}.$$

Случай 2.

$$\left.\begin{array}{l} S_{11} = \{v_{11}, v_{13}, \ldots, v_{1,idx(1,1)}, m_{11}\} \\ S_{21} = \{v_{11}, v_{23}, \ldots, v_{2,idx(2,1)}, m_{21}\} \\ \ldots \\ S_{k-1,1} = \{v_{11}, v_{k-1,2}, \ldots, v_{k-1,idx(k-1,1)}, m_{k-1,1}\} \\ \\ S_{k1} = \{v_{11}, v_{12}\} \\ \\ S_{k+1,1} = \{v_{11}, v_{k+1,2}, \ldots, v_{k+1,idx(k+1,1)}, m_{k+1,1}\} \\ \ldots \\ S_{n(S_1),1} = \{v_{11}, v_{n(S_1),2}, \ldots, v_{n(S_1),idx(n(S_1),1)}, m_{n(S_1),1}\} \end{array}\right\},$$

$$\left.\begin{array}{l} S_{12} = \{v_{21}, v_{13}, \ldots, v_{1,idx(1,1)}, m_{11}\} \\ S_{22} = \{v_{21}, v_{23}, \ldots, v_{2,idx(2,1)}, m_{21}\} \\ \ldots \\ S_{k-1,2} = \{v_{21}, v_{k-1,2}, \ldots, v_{k-1,idx(k-1,1)}, m_{k-1,1}\} \\ S_{k+1,2} = \{v_{21}, v_{k+1,2}, \ldots, v_{k+1,idx(k+1,1)}, m_{k+1,1}\} \\ \ldots \\ S_{n(S_2),2} = \{v_{21}, v_{n(S_1),2}, \ldots, v_{n(S_1),idx(n(S_1),1)}, m_{n(S_1),1}\} \end{array}\right\}.$$

Здесь функция $idx(k,i)$ возвращает максимальное значение второго индекса при v в заданной последовательности S_{ki}, а $n(S_2) = n(S_1) - 1$.

Замечание. С учетом возможного наличия конверсивов слова v_{21} применительно как к обоим указанным случаям РПЗ предполагается, что соответствующая замена уже выполнена, а S_1 и S_2 описывают одно и то же множество объектов относительно одной и той же ситуации, обозначаемой посредством v_{21}, то есть без мены ролей.

Для использования в дальнейших рассуждениях введем функцию $Spv:(v_{11},v_{12})\rightarrow v_{21}$, которая ставит в соответствие расщепленному предикатному значению $\{v_{11},v_{12}\}$ его однословное выражение v_{21}.

Множество РПЗ, определяемых *Утверждением 4.3*, включает в себя расщепления с глаголом-связкой, а также расщепления с глаголами - синтаксическими оформителями ситуаций, обозначаемых именами существительными, и представляющими собой языковое обозначение ролей участников ситуаций.

Обобщая введенное формальное определение РПЗ, дадим теперь понятие конверсива, опираясь на описанные И.А. Мельчуком правила синонимических преобразований типа конверсивных замещений [45, стр. 152-153].

Пусть S_1 и S_2 – пара множеств последовательностей вида (4.1).

Утверждение 4.4. Применительно к $\{S_1,S_2\}$ имеет место конверсив, если для $\forall\, S_{k1}\in S_1$ найдется последовательность $S_{j2}\in S_2$, такая, что при этом могут иметь место следующие случаи взаимного соответствия S_{k1} и S_{j2}.

Случай 1.

$$S_{k1}=\left\{v_{11}{}',v_{k2},v_{k3},\ldots,v_{k,idx(k,1)},m_{k1}\right\},$$

$S_{j2} = \{v_{21}', v_{k2}', v_{k3}, \ldots, v_{k,idx(k,1)}, m_{k1}\}$.

При этом $norm(v_{11}') = norm(v_{21}')$, $norm(v_{k2}) = norm(v_{k2}')$, причем в общем случае $prep(v_{11}') \neq prep(v_{21}')$, а $case(v_{k2}) \neq case(v_{k2}')$.

Случай 2.

$S_{k1} = \{v_{11}', v_{12}', v_{k2}, v_{k3}, \ldots, v_{k,idx(k,1)}, m_{k1}\}$,

$S_{j2} = \{v_{21}', v_{k2}', v_{k3}, \ldots, v_{k,idx(k,1)}, m_{k1}\}$.

Здесь $norm(v_{k2}) = norm(v_{k2}')$, $case(v_{k2}) \neq case(v_{k2}')$ (в общем случае), но при этом для S_{j2} $\exists S_{k1}' \in S_1, S_{k1}' \neq S_{k1}$: $\{S_{k1}', S_{j2}\}$ соответствует *Случаю 1*, а для S_{k1} $\exists S_{j2}' \in S_2$, $S_{j2}' \neq S_{j2}$: $\{S_{k1}, S_{j2}'\}$ также удовлетворяет требованию *Случая 1* настоящего *Утверждения*.

Замечание. Положим $v_{21} = norm(v_{21}')$ в S_{j2} для *Случая 1* и *Случая 2*, $v_{11} = norm(v_{11}')$ и $v_{12} = norm(v_{12}')$ в S_{k1} для *Случая 2*, соответственно. По аналогии с РПЗ будем называть пару $\{v_{11}, v_{12}\}$ расщепленным конверсивом для v_{21}.

Определяемые *Утверждением 4.4* конверсивные замены включают в себя как простые перестановки актантов исходного слова на другие места без расщепления последнего, так и замены РПЗ на их нерасщепленные семантические эквиваленты с последующей перестановкой актантов. В частности, в качестве замен без расщепления могут быть рассмотрены синонимические замещения. Здесь для *Случая 1* мы имеем: $k = j$, $prep(v_{11}') = prep(v_{21}')$, а $case(v_{k2}) = case(v_{k2}')$. *Актуальной* здесь *является* автоматическая лингвистически интерпретируемая классификация выявляемых

конверсивов и определение порядка их замен в анализируемых текстах.

Для установления порядка применения конверсивных преобразований воспользуемся следующими эвристическими правилами.

Правило 1. При выборе возможного варианта конверсивной замены без расщепления предпочтение отдается слову с минимальной многозначностью. При этом степень многозначности количественно определяется числом найденных для рассматриваемого слова предикатных лексических значений.

Правило 2. При нескольких вариантах замен на слова с одинаковым количеством возможных предикатных лексических значений предпочтение отдается слову с максимальным количеством беспредложных валентностей.

Замечание. Как отметил академик Ю.Д. Апресян в [3, стр. 149], беспредложные падежи выступают в качестве обязательных чаще, чем предложные, прямой – чаще, чем косвенные. Данный факт дает основание предположить о том, что из конверсивного ряда более компактное описание ситуации (более четкое выражение смысла) характерно для того предикатного слова, у которого количество беспредложных валентностей максимально.

Правило 3. При наличии нескольких вариантов замены расщепленного конверсива нерасщепленным семантическим эквивалентом следует руководствоваться *Правилом 1* и *Правилом 2* для конверсивных замен без расщепления.

Правило 4. Если для найденного по *Правилу 3* семантического эквивалента расщепленного конверсива существуют вариант замены по *Правилу 1*, либо *Правилу 2*, то следует производить замену расщепленного конверсива именно на этот вариант.

Для решения задачи лингвистически интерпретируемой классификации конверсивов, выявляемых в соответствии с *Утверждением 4.4* на основе вышеуказанных *Правил 1-4*, будем использовать уже рассмотренные методы АФП.

Введем в рассмотрение формальный контекст:

$$K^{Conv} = \left(G^{Conv}, M^{Conv}, I^{Conv}\right), \tag{4.5}$$

в котором согласно *Утверждению 4.4*

$$G^{Conv} = \left\{v_{21} : v_{21} = norm\left(v_{21}{}'\right)\right\},$$

$$M^{Conv} = \left\{v^{Conv} : v^{Conv} = \begin{cases} v_{11} \\ v_{12} \bullet ":" \bullet v_{11} \end{cases}\right\},$$

где $v_{11} = norm\left(v_{11}{}'\right)$;

$v_{12} = norm\left(v_{12}{}'\right)$;

операция конкатенации имеет место для *Случая 2* из рассматриваемых *Утверждением 4.4*;

отношение $I^{Conv} \subseteq G^{Conv} \times M^{Conv}$ ставит в соответствие каждому варианту конверсивной замены $v_{21} \in G^{Conv}$ заменяемый конверсив $v^{Conv} \in M^{Conv}$.

Пусть $\Re^{Conv}$ есть решетка ФП для контекста (4.5). Введем индексы: 1 – для контекстов вида (3.12) и (4.5), формируемых с применением предложенной нами методики выделения и классификации синтаксических отношений, рассмотренной в *Разделе 3.5*; 2 – для контекстов тех же видов, но формируемых на основе синтаксического разбора ЕЯ-фраз программой “Cognitive Dwarf”. Положим, что решетки $\Re_2^{Conv}$ и $\Re_2^{F}$ формируются на основе неструктурированного текста заданной тематики, включающего подмножество множества T относительно языкового контекста

ситуации (1.1). Мощность этого подмножества зависит от репрезентативности текста. Под показателем репрезентативности здесь следует понимать количество форм языкового описания заданной ситуации, присутствующих в анализируемом тексте и использованных при формировании $\Re_1^F$ и $\Re_1^{Conv}$.

Каждая область решетки $\Re^{Conv}$ (вне зависимости от исходных данных для построения) при единственности НОПП и НОСП получает содержательную интерпретацию группы смысловых отношений со сходным составом аргументов и сходным характером перестановок аргументов (типом конверсии).

Введем в рассмотрение базисы импликаций: L_1^{Conv} – базис импликаций для контекста K_1^{Conv}, L_2^{Conv} – для контекста K_2^{Conv}, соответственно.

Утверждение 4.5. Будем считать классификацию отношений из R в (1.1) на основе контекста (3.12) допустимой применительно к случаю наличия в T фраз, отвечающих условиям *Утверждения 4.4*, если $\Re_1^F \subset \Re_2^F$ и $\exists\left(\mathrm{Pr}_1^{Conv} \to Cs_1^{Conv}\right) \in L_1^{Conv} : \exists\left(\mathrm{Pr}_2^{Conv} \to Cs_2^{Conv}\right) \in L_2^{Conv}$, где $\mathrm{Pr}_1^{Conv} \cap \mathrm{Pr}_2^{Conv} \neq \varnothing$ и $Cs_1^{Conv} \cap Cs_2^{Conv} \neq \varnothing$.

При этом случай $\Re_1^F = \Re_2^F$ не обязательно соответствует тексту с максимальной репрезентативностью по сформулированному нами критерию. Встречаемость тех или иных сочетаний флексий находится в зависимости и от количества описываемых текстом ситуаций. В частности, текстом может описываться несколько ситуаций, близких рассматриваемой по составу участников и их ролевой ориентации. Вопросам взаимосвязи качественных характеристик решеток ФП и

информативности отдельного признака в текстовой классификации посвящается следующий раздел.

4.4. Информативность признака и критерий полезности решетки формальных понятий

Используемое для формирования моделей (4.4) и (4.5) множество текстов представляет собой тематическое подмножество того текстового корпуса, который по жанровому разнообразию представленного в нем рода словесности [77] следует отнести к научной прозе. Рассмотрим, каким образом особенности исходных текстов влияют на качество концептуальной кластеризации, выполняемой методами АФП.

Вначале сформулируем более общее определение понятия репрезентативности, введенного нами в предыдущем разделе.

Определение 4.1. Под репрезентативностью множества текстов будем понимать способность этого множества отображать все свойства Предметной Области, релевантные для некоторого заданного лингвистического исследования.

При использовании последовательностей вида (4.1) в качестве основы кластеризации выбираемая оценка репрезентативности для исходного текстовго материала должна стать основой практических выводов как оносительно точности алгоритмов синтаксического анализа, так и направлениях их дальнейшего совершенствования. В этом плане естественной оценкой репрезентативности может послужить суммарная частота Fs, с которой последовательности вида (4.1), соответствующие условию *Утверждения 4.1*, встречаются в анализируемых текстах. Но с учетом отсутствия ограничений на тип q отношения R_q между словами в (4.1) за указанную оценку следует

принять отношение частоты Fs к количеству n_q типов отношений R_q в рамках последовательностей вида (4.1):

$$Fq = \frac{Fs}{n_q} = \frac{n_S}{nn_q}, \qquad (4.6)$$

где n_S есть количество последовательностей вида (4.1), извлеченных из анализируемого множества текстов; n есть общее количество слов в анализируемом множестве текстов.

Хорошим примером репрезентативности текста в соответствии с критерием (4.6) с характерной минимизацией n_q при максимизации Fs может послужить обзорная статья [5]. На рис. 4.1 представлена решетка ФП для указанного текста. Соответствующий ей формальный контекст $K^V = \left(G^V, M^V, I^V\right)$ можно представить как получаемый из формального контекста вида (4.4), в котором $G = \left\{g^V\right\}$, где g^V есть некоторая пометка для рассматриваемого текста. При этом

$$G^V = \left\{m \in M : \exists v \in V, \left(g^V, m, v\right) \in I\right\},$$

$$M^V = \left\{v \in V : \exists m \in M, \left(g^V, m, v\right) \in I\right\},\ I^V = \left\{(m, v) : \left(g^V, m, v\right) \in I\right\}.$$

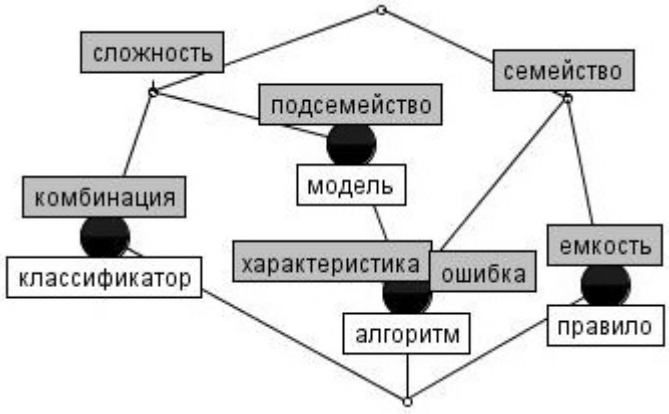


Рис. 4.1. Пример решетки ФП для множества ситуационных контекстов

Репрезентативность текстового материала в значительной мере влияет на способность решетки ФП выделять общие свойства классифицируемых объектов и соответствие формируемой решетки требованию иерархичности лексических ресурсов.

С целью достижения указанных требований для решетки в работе [124] был предложен критерий полезности. Если A_i – объем, B_i – содержание формального понятия (A_i, B_i) согласно *Определению 1.10*, то данный критерий следует рассматривать как коэффициент F:

$$F = \max_{j=1}^{J} \left(\sum_{i=1}^{n_j} |A_i| \right), \qquad (4.7)$$

где J – индексное множество цепочек; $j \in J$ - номер цепочки; n_j – количество ФП в цепочке с номером j; i – порядковый номер ФП в цепочке.

Максимизация указанного критерия при генерации формального контекста вида (4.5), в частности, предполагает выбор пар $\{v_{21}, v^{Conv}\}$ таким образом, чтобы любое ФП $C^{Conv} = \left(A^{Conv}, B^{Conv}\right)$ в решетке $\Re^{Conv}\left(G^{Conv}, M^{Conv}, I^{Conv}\right)$ входило в цепочку максимальной длины при $\left|A^{Conv}\right| \to \max$.

При этом само формирование решетки ведется по областям. Вначале на основе групп подряд идущих последовательностей вида (4.1) на выходе синтаксического анализа *Алгоритмом 4.3* выявляются пары соподчиненных слов, задающих РПЗ и расщепленные конверсивы в соответствии с условиями *Утверждений 4.3* и *4.4*. Этим же алгоритмом производится замена найденных РПЗ и конверсивов на их однословные выражения согласно *Правилам 1-4* во всех исходных последовательностях соподчиненных слов для последующего

использования указанных последовательностей в качестве исходных данных *Алгоритма 4.1*. Функция $Conv: v^{Conv} \to v_{21}$, упоминаемая в *Алгоритме 4.3*, есть обобщение функции $Spv: (v_{11}, v_{12}) \to v_{21}$, введенной нами ранее для расщепленных предикатных значений, выявляемых в соответствии с *Утверждением 4.3*. При этом

$$v^{Conv} = \begin{cases} v_{11} \\ v_{12} \bullet ":" \bullet v_{11} \end{cases} \tag{4.8}$$

согласно разделению множества признаков формального контекста вида (4.5).

Алгоритм 4.3. Формирование кандидатов на включение в отношение I^{Conv}.

Вход: $P^S = \left\{ P_i^S : P_i^S = \left\{ S_{ki} : S_{ki} = \left\{ v_1, \ldots, v_{n(k,i)}, m_{ki} \right\} \middle| \; i = \overline{1, n(G)} \right\} \right\}$;

Выход: P^C ; // Множество объектов с наборами признаков

$P^{Conv} = \left\{ \left(v^{Conv}, v_{21} \right) : v_{21} = Conv\left(v^{Conv} \right) \right\}$;

P^{SC} ; // Множество, полученное заменой РПЗ и конверсивов во всех $S_{ki} \in P_i^S$ из исходного P^S

Начало

$P^C := \varnothing$; $P^{Conv} := \varnothing$; // Инициализация

Начало цикла. Для $i = 1, \ldots, n(G)$

Сформировать множество $P_i^{S'}$ из групп $P_{ki}^S \subseteq P_i^S$ подряд идущих S_{ki} с одним и тем же v_1;

Конец цикла {Для $i = 1, \ldots, n(G)$ *}*;

$P^{S'} := \left\{ P_i^{S'} \middle| i = \overline{1, n(G)} \right\}$;

Начало цикла. Для всех $P_i^{S'}$ таких, что $i = \overline{1, n(G)}$

Выбрать $P_j^{S'} \in P^{S'}: \ j \neq i$;

Начало цикла. Для всех $P_{k1i}^{S} \in P_i^{S'}$

Найти $P_{k2i}^{S} \in P_j^{S'}: \left\{P_{k1i}^{S}, P_{k2j}^{S}\right\}$ удовлетворяет условию

Утверждения 4.4;

$P^{Conv} := P^{Conv} \cup \left\{\left\langle v^{Conv}, v_{21}\right\rangle\right\}$ согласно (4.8);

Если $P^{C} = \varnothing$ *то*

$P_k^{C} := \left\{v^{Conv}\right\}$;

$P^{C} := P^{C} \cup \left\{\left\langle v_{21}, P_k^{C}\right\rangle\right\}$;

иначе

Найти $\left\langle v_{21}, P_k^{C}\right\rangle \in P^{C}$;

$P^{C} := P^{C} \setminus \left\{\left\langle v_{21}, P_k^{C}\right\rangle\right\}$;

$P_k^{C} := P_k^{C} \cup \left\{v^{Conv}\right\}$;

$P^{C} := P^{C} \cup \left\{\left\langle v_{21}, P_k^{C}\right\rangle\right\}$;

Конец {Если $P^{C} = \varnothing$ *}*;

Конец цикла {Для всех $P_{k1i}^{S} \in P_i^{S'}$ *}*;

Конец цикла {Для всех $P_i^{S'}$ *таких, что* $i = \overline{1, n(G)}$ *}*;

$P^{SC} := \varnothing$;

Начало цикла. Для всех P_i^{S} таких, что $i = \overline{1, n(G)}$

$P_i^{SC} := \varnothing$;

Начало цикла. Для всех $S_{ki} \in P_i^{S}$

Сформировать S_{ki}^{SC} заменой v^{Conv}: $\exists\left(v^{Conv}, v_{21}\right) \in P^{Conv}$

на v_{21} в S_{ki} согласно *Правилам 1-4*;

$P_i^{SC} := P_i^{SC} \cup \left\{S_{ki}^{SC}\right\}$;

Конец цикла {Для всех $S_{ki} \in P_i^S$ *}*;

$P^{SC} := P^{SC} \cup \left\{P_i^{SC}\right\}$;

Конец цикла {Для всех P_i^S *таких, что* $i = \overline{1, n(G)}$ *}*;

Конец {Алгоритм 4.3}.

Отдельная цепочка $P_{Ch(j)}^C$, дополненная соседними ФП, формируется на основе множества P^C объектов с заданными наборами признаков согласно *Алгоритму 4.4*. С целью минимизации числа спорных ФП каждое следующее ФП в цепочке выбирается по принципу постепенного уменьшения содержания и максимизации количества общих признаков с потенциальным подпонятием при минимуме общих признаков с любым ФП, не входящим в цепочку.

Алгоритм 4.4. Формирование цепочки в $\Re^{Conv}$ по максимуму критерия (4.7).

Вход: P^C на выходе *Алгоритма 4.3*;

Выход: $P_{Ch(j)}^C = \left\{\left(v_{21}, P_k^C\right) \middle| \left(v_{21}, P_k^C\right) \in P^C, \leq\right\}$;

// P_k^C – набор признаков для v_{21}

P^R; // Подмножество исходного P^C, не вошедшее в $P_{Ch(j)}^C$

$P_{Neigh(j)}^C \subset P_{Ch(j)}^C$; // Соседние ФП для тех, относительно

// которых рассматривается отношение $\leq$

Начало

$P^C_{Ch(j)} := \varnothing$;

$P^C_{Neigh(j)} := \varnothing$; // Инициализация

Выбрать $\left(v_{\max}, P^C_{\max}\right)$ из $P^C : \left|P^C_{\max}\right| \to \max$;

$P^C := P^C \setminus \left\{\left(v_{\max}, P^C_{\max}\right)\right\}$;

$P^C_{Ch(j)} := P^C_{Ch(j)} \cup \left\{\left(v_{\max}, P^C_{\max}\right)\right\}$;

$P^C_{tmp} := P^C_{\max}$;

Начало цикла

Выбрать $\left(v_{21}, P^C_k\right)$ из $P^C : P^C_k \subset P^C_{tmp}$ и

$$\left|P^C_{tmp} \cap P^C_k =: Cr\right| \to \max;$$

При $Cr = \varnothing$ *выход из цикла*;

$P^C_{tmp} := P^C_k$;

$P^C_{Ch(j)} := P^C_{Ch(j)} \cup \left\{\left(v_{21}, P^C_k\right)\right\}$;

$P^C := P^C \setminus \left\{\left(v_{21}, P^C_k\right)\right\}$;

Выбрать $\left\{\left(v_{Cr}, P^C_{Cr}\right) \middle| P^C_{Cr} \supseteq Cr\right\} =: P^{Cr} \subseteq P^C$;

$P^C_{Ch(j)} := P^C_{Ch(j)} \cup P^{Cr}$;

$P^C_{Neigh(j)} := P^C_{Neigh(j)} \cup P^{Cr}$;

$P^C := P^C \setminus P^{Cr}$;

Конец цикла;

$P^R := P^C$;

Конец {Алгоритм 4.4}.

Алгоритмом 4.5 строится множество цепочек для множества $P^C_{Neigh(j)} \subset P^C_{Ch(j)}$. Множество $P^C_{Neigh(j)}$ есть в соответствии с *Определением 1.17* множество ФП, соседних по отношению к тем ФП $C^{Conv} = \left(A^{Conv}, B^{Conv}\right)$: $A^{Conv} = \{v_{21}\}$, $B^{Conv} = P^C_k$, между которыми устанавливается отношение $\leq$ при формировании цепочки.

Алгоритм 4.5. Генерация множества цепочек для “соседних” ФП в решетке $\Re^{Conv}$.

Вход: P^C на выходе *Алгоритма 4.3*;

Выход: $P^C_{Ch} = \left\{P^C_{Ch(j)} : P^C_{Ch(j)} = \left\{\left(v_{21}, P^C_k\right) : \left(v_{21}, P^C_k\right) \in P^C\right\}\right\}$;

Начало

$P^C_{Ch} := \varnothing$; // Инициализация

Начало цикла

Сформировать $P^C_{Ch(j)}$, $P^C_{Neigh(j)}$ и P^R *Алгоритмом 4.4* на основе P^C;

При $\left|P^C_{Ch(j)}\right| \leq 1$ *выход из цикла*;

$P^C_{Ch} := P^C_{Ch} \cup \left\{P^C_{Ch(j)}\right\}$;

$P^C := P^C_{Neigh(j)} \cup P^R$;

Конец цикла;

Конец {Алгоритм 4.5}.

Немаловажную роль при максимизации критерия (4.7) для решетки ФП играет инофрмативность каждого признака. Как было показано в [124], информативность признака тем ниже, чем большим

количеством объектов рассматриваемого формального контекста он разделяется.

При построении $\mathfrak{R}^{Conv}$ с применением *Алгоритмов 4.3-4.5* значимость неинформативных признаков будет минимальной согласно *Правилу 1* порядка применения конверсивных преобразований (доказательство очевидно). Поэтому $K^{Conv} := \bigcup_{j=1}^{J} P_{Ch(j)}^{C}$ на выходе *Алгоритма 4.5*.

На рис. 4.2 представлен пример решетки $\mathfrak{R}^{Conv}$, построенной с применением *Алгоритмов 4.3-4.5*. В качестве экспериментального текстового материала были взяты варианты ответов на тестовые задания открытой формы по материалам статьи [5]. Область в решетке, отвечающая условию *Утверждения 4.5*, обозначена прямоугольником. Для сравнения на рис. 4.3 показана аналогичная решетка, полученная для примера из *Таблицы 3.2* в соответствии с *Теоремой 3.1*.

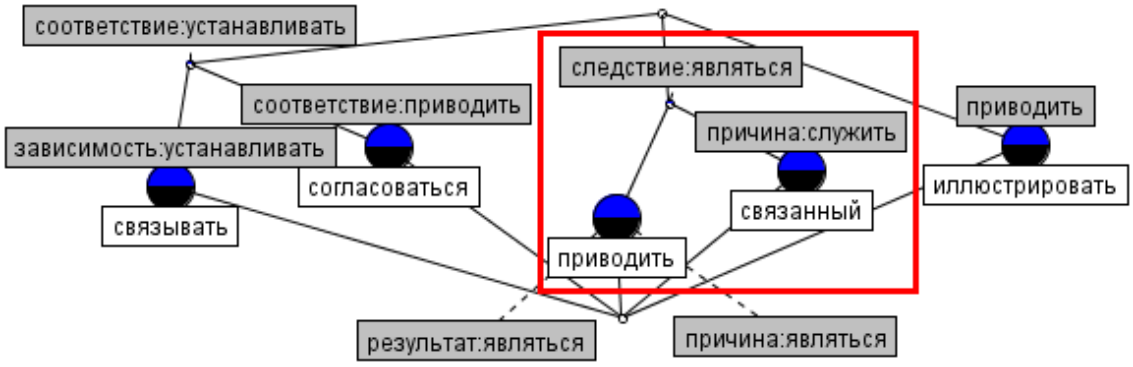


Рис. 4.2. Группировка РПЗ и конверсивных замен по результатам Cognitive Dwarf

Рассмотрим теперь решетку $\mathfrak{R}^{V}\left(G^{V}, M^{V}, I^{V}\right)$ для множества ситуационных контекстов вида (4.1), пример которой представлен на рис. 4.1, в плане максимизации критерия (4.7).

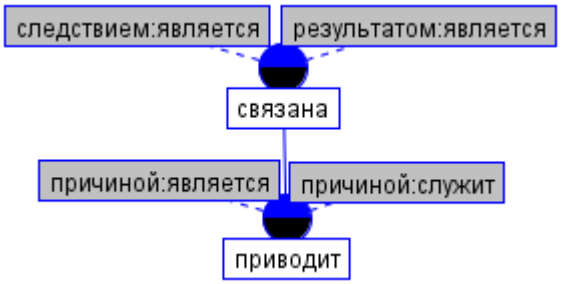


Рис. 4.3. РПЗ и конверсивы в составе фраз из T' (табл. 3.1)

При отборе признаков, которыми будут характеризоваться объекты в составе множества G^V, в целях минимизации влияния неинформативных признаков на вычисляемое значение критерия (4.7) для решетки $\Re^V$ следует учитывать частоту $Cnt(v)$, с которой в анализируемом тексте потенциальный признак v встречается с различными $m \in G^V$.

Пусть P^{Cnt} есть множество пар вида $(v, Cnt(v))$ для каждого признака множества M^V. Положим, что множество P^{CV} есть аналог множества P^C на выходе *Алгоритма 4.3* и содержит пары вида “объект – набор признаков” для формального контекста $K^V = (G^V, M^V, I^V)$. Введем также в рассмотрение P_{Ch}^{CV} – аналог множества P_{Ch}^C, формируемого *Алгоритмом 4.5*. Тогда формирование контекста K^V с исключением из рассмотрения малоинформативных признаков можно представить с помощью следующего алгоритма.

Алгоритм 4.6. Генерация формального контекста K^V.

Вход: $P_i^S = \{S_{ki} : S_{ki} = \{v_1, \ldots, v_{n(k,i)}, m_{ki}\}\}$;

Выход: $K^V = (G^V, M^V, I^V)$;

Начало

Сформировать P^{CV} на основе P_i^S;

Сформировать P^{Cnt};

$\Delta_F := 0$;

Начало цикла. Пока $\Delta_F \le 0$

$\Delta_F := |\Delta_F|$;

Сформировать P_{Ch}^{CV} на основе P^{CV};

$F_{tmp} := \max_{j=1}^{J^V}\left(\left|P_{Ch(j)}^{CV} : P_{Ch(j)}^{CV} \in P_{Ch}^{CV}\right|\right)$;

// J^V – индексное множество цепочек относительно

// решетки $\Re^V$

$\Delta_F := \Delta_F - F_{tmp}$;

Найти $v \in M^V : (v, Cnt(v)) \in P^{Cnt}$ и $Cnt(v_C)$ – максимально;

Начало цикла. Для всех $\left(m, P_k^{CV}\right) \in P^{CV}$

$P_k^{CV} := P_k^{CV} \setminus \{v\}$;

Конец цикла {Для всех $\left(m, P_k^{CV}\right) \in P^{CV}$ *}*;

$P^{Cnt} := P^{Cnt} \setminus \{(v, Cnt(v))\}$;

Конец цикла {Пока $\Delta_F \le 0$ *}*;

$K^V := \bigcup_{j=1}^{J^V} P_{Ch(j)}^{CV}$;

Конец {Алгоритм 4.6}.

Следует отметить, что зависимость вероятности, с которой подпоследовательность слов из структуры (4.1), выделяемая согласно *Алгоритму 4.1* при формировании пар “объект-признак”, будет

подчиняться некоторому другому слову этого же синтаксического контекста в рассматриваемом корпусе текстов, от вероятностей появления в корпусе этого слова и подпоследовательности отдельно друг от друга *Алгоритмом 4.6* не учитывается. Причина заключается во взаимной зависимости составов таких подпоследовательностей, вытекающей из *Утверждения 4.2*, при их употреблении в тексте за рамками синтаксического контекста (4.1). Использование мер информативности различных комбинаций слов из (4.1) с учетом указанной зависимости, а также отсутствия ограничений на тип синтаксического отношения между соподчиненными словами – тема отдельного прикладного исследования.

Выводы

Предложенный в настоящей главе комплексный подход к решению задачи кластеризации текстов основан на соотношении смыслов соподчиненных слов в составе синтаксического контекста имени существительного. При этом рассмотренные в главе ситуации частичной смысловой эквивалентности подтверждают полученный нами в первой главе вывод о синтаксических отношениях как частном случае семантических отношений, а также возможности выделения и кластеризации самих семантических отношений по результатам синтаксического анализа текстов заданного тематического корпуса. При использовании последовательностей соподчиненных слов как основы выявления расщепленных значений решетка ФП для совокупности РПЗ, в частности, позволяет выделять группы смысловых отношений из задаваемых ЛФ-параметрами.

Наряду с выделением семантических отношений, рассмотрение синтаксического контекста существительного в качестве базовой

структуры семантической кластеризации позволяет решить задачу автоматического извлечения элементов толкования лексического значения непосредственно из текстов. Сказанное дает возможность формирования прецедентов для ситуаций ЛФ-синонимии также на основе множеств текстов, в каждом из которых все тексты семантически эквивалентны друг другу.

Применительно к множеству выявляемых синтаксических контекстов существительных рассмотренный в заключительном разделе главы критерий полезности решетки ФП позволяет делать выводы о силе семантической связи слов в рамках указанных контекстов. К примеру, чем в большем количестве синтаксических контекстов фигурирует заданное предикатное слово, тем менее однозначно оно определяет существительное, ему подчиненное, и, следовательно, тем меньше сила их семантической связи [124], что означает и меньшее значение полезности решетка для множества ситуационных контекстов в соответствии с *Алгоритмом 4.6*.

Значение критерия полезности решетки ФП для совокупности РПЗ дает возможность делать выводы о сходстве ролевого состава ситуаций, обозначаемых в составе расщепленных предикатных значений словами-аргументами той или иной лексической функции.

В следующей главе мы рассмотрим, каким образом на основе синтаксического контекста имени существительного вычисляется количественная мера схожести ситуаций языкового употребления, порождаемых независимо друг от друга, а также перспективы использования указанного контекста в задаче сжатия информации при построении текстовых баз данных по заданной предметной области.

Глава 5

МЕТОДЫ НАХОЖДЕНИЯ СЕМАНТИЧЕСКОГО РАССТОЯНИЯ МЕЖДУ ТЕКСТАМИ ПРЕДМЕТНОГО ЯЗЫКА

В данной главе рассматриваются вопросы использования мер близости в решетках формальных понятий применительно к формализованному описанию текстов формальными контекстами для ситуаций языкового употребления. Описывается построение формального контекста ситуации языкового употребления на основе множества семантически эквивалентных фраз предметно-ориентированного подмножества естественного языка. Излагается метод редукции формального контекста удалением информации расщепленных предикатных значений. Рассматривается модель тезауруса предметной области в виде формального контекста для совокупности ситуаций употребления заданного предметно-ориентированного языкового подмножества и ориентированная на нее модель отдельной ситуации в виде объекта с заданным набором признаков. Вводится мера схожести между формальными контекстами ситуаций языкового употребления. Описываются правила установления семантической эквивалентности фраз предметно-ориентированного подмножества естественного языка.

5.1. Синтаксические и семантические связи в ситуации языкового употребления

В *Разделе 3.5* нами было рассмотрено выделение и классификация синтагматических зависимостей на основе множества СЭ-фраз. Предположим теперь, что элементами множества R в

модели (1.1) являются произвольные отношения между объектами $o \in O$. Кроме того, мы расширим возможности синонимического варьирования для T, введя синонимию на уровне предметной лексики наряду с лексико-функциональной.

Дадим содержательное описание тех изменений, которые необходимо внести в модель процесса формирования множества R.

При рассмотрении задачи выделения и классификации синтаксических отношений в качестве основы формирования R относительно структуры (1.1) мы брали множество неизменных частей всех слов, употребленных во всех фразах, представляемых множеством T.

С учетом наличия РПЗ и конверсивов в словесном обозначении самой ситуации S, в роли слов, которые присутствуют во всех фразах синонимического множества, могли выступать только словесные обозначения “участников” ситуации.

Будем рассматривать введенное ранее индексное множество J применительно к неизменным частям всех слов, употребленных в более чем одной ЕЯ-фразе из множества T. При этом удвоенная длина общей неизменной части пары слов всегда больше суммы длин изменяемых (флективных) частей.

Последовательность индексов неизменных частей слов, присутствующих в $T_i \in T$, рассматривалась как модель линейной структуры этой фразы. Обозначим множество указанных моделей на J как L^S. Тогда при наличии синонимов в словесных обозначениях либо участников ситуации S, либо характеристик участников будет справедливы следующие свойства моделей $L(T_i) \in L^S$.

Лемма 5.1. Пара индексов $\{j_1, j_2\} \subset J$ соответствует словам-синонимам, если $\exists \{L(T_1), L(T_2)\} \subseteq L^S : L(T_1) = J_1 \bullet \{j_1\} \bullet J_2$ и

$L(T_2) = J_1 \bullet \{j_2\} \bullet J_2$, где $J_1 \subset J$, $J_2 \subset J$, а "$\bullet$" есть операция типа конкатенации над множеством J.

Доказательство леммы следует из определения, сформулированного нами в *Разделе 3.5* для синтаксической связи применительно к модели линейной структуры предложения.

Пусть P^J – множество пар, отвечающих условию *Леммы 5.1*. Заменим индексы, вошедшие в пары из P^J, на некоторые $j \in (\mathrm{N} \setminus J)$ во всех $L \in L^S$, где N – множество натуральных чисел. Обозначим преобразованное L^S как $L^{S'}$, множество заменяемых индексов – как J^P, а множество индексов, на которые производится замена – как $J^{P'}$, $J^{P'} \cap J^P = \varnothing$. Фактически каждая модель в $L^{S'}$ задается на множестве $\left(J \setminus J^P\right) \cup J^{P'}$.

Теорема 5.1 Справедливым будет утверждать, что индексы с максимальной встречаемостью в различных моделях из множества $L^{S'}$ соответствуют словам-существительным, обозначающим участников ситуации (1.1).

Доказательство теоремы следует из доказанной *Леммы 1* и сделанного допущения о наличии РПЗ и конверсивов в словесных обозначениях ситуаций.

Обозначим множество индексов, удовлетворяющих условию *Теоремы 5.1*, как J^N. Пусть $L_1(T_i) \in L^{S'}$, а $L_2(T_i)$ – модель линейной структуры того же предложения, но относительно J^N. Обозначим множество моделей второго вида как L^N. Положим также, что имеется $L_j^{S'} \subset L^{S'}$ такое, что для всех $L_1(T_i) \in L_j^{S'}$ модели $L_2(T_i)$ одинаковы и соответствуют некоторой $L_2(T_j) \in L^N$, $T_j \in T$.

Теорема 5.2 Индексы $j \notin J^N$ с максимальной частотой встречаемости в различных моделях $L_1(T_i) \in L_j^{S'}$ соответствуют либо словам-наречиям, либо прилагательным, либо опорным существительным в составе генитивных конструкций.

Доказательство. Исключением из множества $L_j^{S'}$ тех моделей, все индексы в составе которых входят в J^N, с последующим удалением индексов $j \in J^N$ из оставшихся моделей, получаем частный случай *Теоремы 5.1*.

Обозначим множество индексов, удовлетворяющих условию *Теоремы 5.2*, как J^A. Установление синтаксических ролей и выделение флексий для слов с индексами из $\left(\left(J \setminus J^P\right) \cup J^{P'}\right) \setminus \left(J^N \cup J^A\right) \cup \{0\}$ производится по аналогии с выявлением указанной информации у слов в составе РПЗ описанным в *Разделе 3.5* способом. При этом вместо индексов с ненулевым значением рассматриваются индексы из $J^N \cup J^A$.

Таким образом, в соответствии с требованием иерархичности знаний о синонимии множество R отражает:

– сочетаемость основ синтаксически главных и зависимых слов. Данный вид отношений необходим для выделения объектов и признаков во всех рассматриваемых видах синонимии;

– сочетаемость флексий главных и зависимых слов. Фактически здесь задаются значения признаков для классов СЭ;

– сочетаемость слова и его лексико-семантических производных в рамках РПЗ. Указанные отношения значимы для выделения и классификации случаев лексико-функциональной синонимии.

Сами семантические отношения при этом составляют основу классификации и вычисления меры схожести ситуаций употребления ЕЯ.

5.2. Формальный контекст ситуации языкового употребления и методы его построения

Задача классификации и анализа схожести ситуаций употребления ЕЯ наиболее естественно решается методами АФП, рассмотренными в предыдущих главах.

Отметим особенности объектов и признаков для отдельной ситуации языкового употребления, представляемой моделью вида (1.1), и для совокупности таких ситуаций, подлежащих сравнению.

Множество объектов G^S формального контекста

$$K^S = \left(G^S, M^S, I^S\right) \qquad (5.1)$$

одной ситуации составляют основы слов, входящих во фразы из множества T и являющихся зависимыми по отношению к другому слову из некоторой ЕЯ-фразы $T_i \in T$.

Множество признаков M^S включает в себя подмножества, обозначаемые далее посредством M с соответствующим нижним индексом и содержащие:

- указания на основу синтаксически главного слова (M_1);
- указания на флексию главного слова (M_2);
- связи "основа-флексия" для синтаксически главного слова (M_3);
- сочетания флексий зависимого и главного слова (M_4). При этом после флексии главного слова через двоеточие

указывается предлог (если такой имеется) для связи главного слова с зависимым;

– указания на флексию зависимого слова (M_5).

Посредством $I^S \subseteq G^S \times M^S$ отношения из множества R разбиваются на классы по сходству:

– основы главного слова, что особенно актуально для исследования сочетаемости в рамках ЛФ-параметров, посредством которых описываются РПЗ;

– флексии зависимого слова, что необходимо для выделения и обобщения синтаксических отношений;

– лексической и флективной сочетаемости, что позволяет выявить зависимости, аналогичные смысловой связи между опорным словом и генитивной именной группой в составе генитивной конструкции русского языка.

При этом каждому классу соответствует некоторое формальное понятие в решетке $\mathfrak{R}^S\left(G^S, M^S, I^S\right)$.

Решетка $\mathfrak{R}^S$ для примера ситуации ЕЯ-употребления, рассмотренного в *Разделе 3.5*, представлена на рис. 5.1. Здесь ранее использованное СЭ-множество дополнено новыми ЕЯ-фразами, полученными из уже имеющихся фраз путем синонимических замен как абстрактных слов и их сочетаний (“*является следствием*” – “*служит причиной*”), так и предметной лексики (“*переобучение*” – “*переподгонка*”). В целях компактности изложения графического материала в формальный контекст не были включены объекты и признаки для прилагательных (“*эмпирический*” и “*нежелательное(ая)*”).

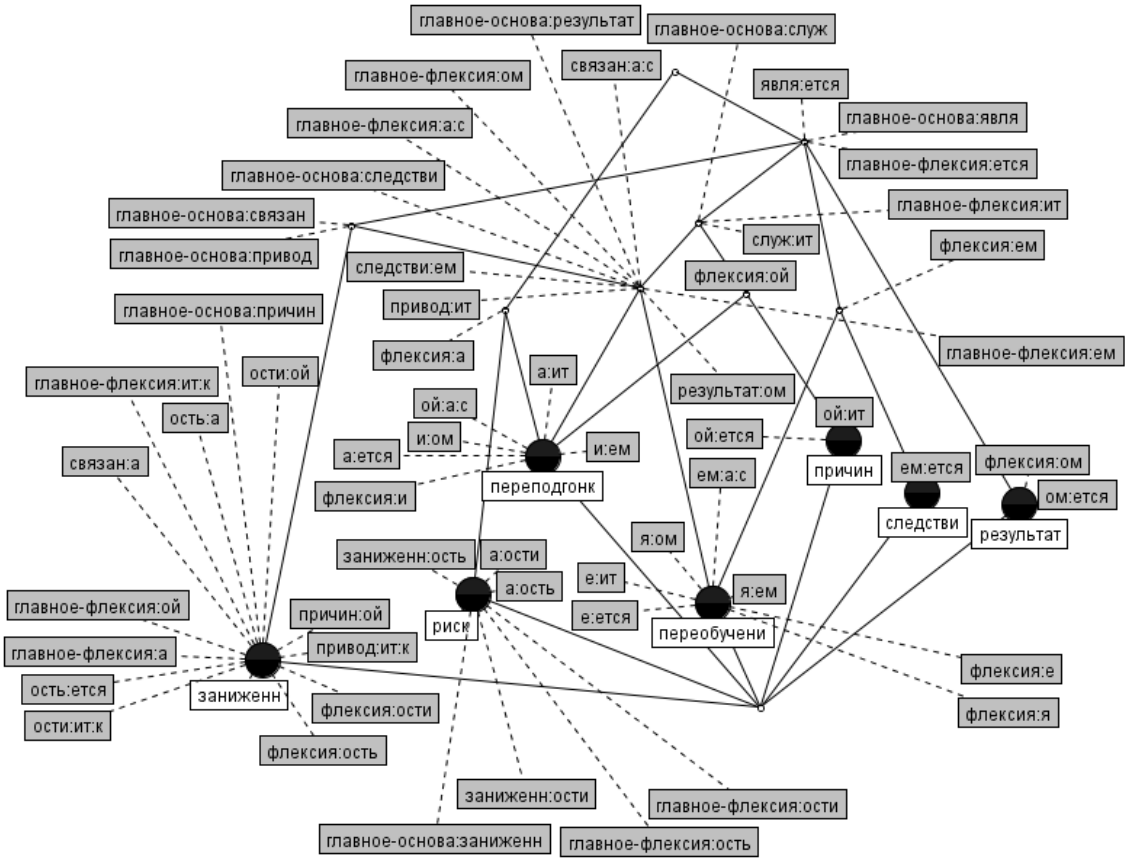


Рис. 5.1. Пример формального контекста ситуации языкового употребления

Классы ФП в решетке различаются степенью абстракции, которая зависит от частоты употребления главных слов анализируемых сочетаний в различных синтаксических контекстах относительно модели (1.1). Для количественной оценки СЭ значимы классы одного уровня абстракции, соответствующие подчинению существительных, обозначающих участников ситуации, тем словам, которые ее называют и не входят в РПЗ. *Необходима* редукция контекста вида (5.1) исключением объектов и признаков РПЗ.

Теорема 5.3. Пусть $\{m_1, m_2, m_3\} \subset M_1^S$. Если считать m_1, m_2 и m_3 взаимно различными, то m_1 соответствует указанию на основу главного, m_2 – зависимого слова РПЗ, а m_3 – указанию на основу однословного эквивалента РПЗ при выполнении трех условий:

1. $\exists g_1 \in G^S : I^S(g_1, m_1) = true$, $I^S(g_1, m_3) = false$, $m_2 = p_{bs} \bullet g_1$. Здесь символ “$\bullet$” обозначает конкатенацию, а p_{bs} есть используемое далее обозначение для символьной константы *“главное-основа:”*.
2. $\exists \{g_2, g_3\} \subset G^S$, при этом объекты g_1, g_2 и g_3 являются взаимно различными, а

$$I^S(g_2, m_3) \wedge I^S(g_3, m_3) \wedge$$

$$\wedge \left(I^S(g_2, m_1) \wedge I^S(g_3, m_2) \vee I^S(g_2, m_2) \wedge I^S(g_3, m_1) \right) = true.$$

3. Не существует других троек объектов, для которых признак m_3 занимал бы место либо признака m_1, либо признака m_2 в вышеуказанных соотношениях.

Доказательство теоремы следует из свойств базиса импликаций для формального контекста вида (5.1).

Исключая объекты и признаки слов расщепленных предикатных значений согласно *Теореме 5.3* для приведенного на рис. 5.1 примера, получаем редуцированный формальный контекст, решетка ФП для которого представлена на рис. 5.2.

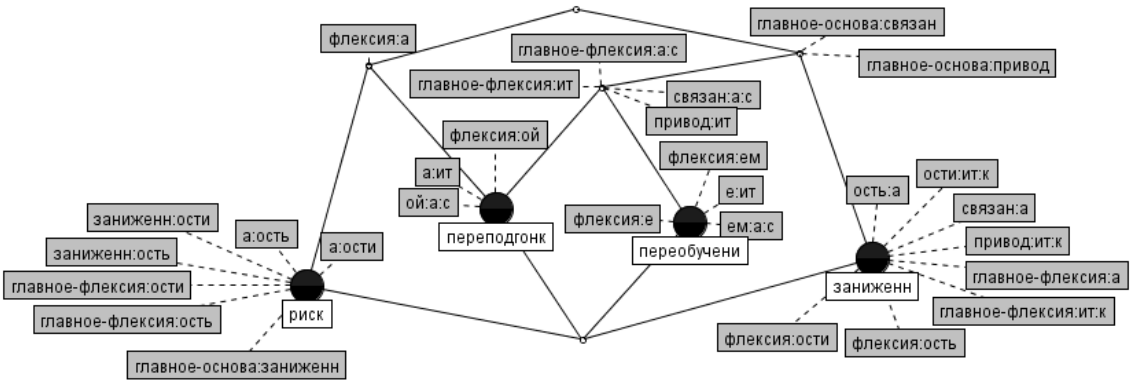


Рис. 5.2. Решетка ФП для редуцированного формального контекста

После удаления информации РПЗ формальный контекст вида (5.1) отражает классы отношений, которые определяются исключительно ролями объектов-участников ситуации по отношению к ней самой. При этом синтаксические зависимости как частный случай семантических отношений выражаются определенными сочетаниями флексий. Сказанное позволяет в ряде случаев выделять основы и их сочетания на базе указанных морфологических зависимостей. Эти зависимости могут быть либо выявлены ранее для других ситуаций языкового употребления, либо найдены с помощью программ синтаксического анализа, реализующих стратегию разбора на основе наиболее вероятных связей слов. Фактически данные связи и выделяет модель, предложенная нами в *Разделе 3.5* и дополненная в настоящей главе.

5.3. Тезаурус предметной области и схожесть ситуаций языкового употребления

Рассмотрим теперь *задачу* накопления и систематизации знаний, представляемых структурами вида (5.1). Если указанные знания формируются на основе независимого ЕЯ-описания различных фактов некоторой предметной области группой экспертов, то получаемая структура будет соответствовать тезаурусу этой предметной области. При этом предполагается, что: (а) из множеств объектов и признаков каждой рассматриваемой ситуации языкового употребления удалена информация расщепленных предикатных значений, (б) выделение самих объектов и признаков производится как на основе модели, предложенной в настоящей работе, так и с помощью известных синтаксических анализаторов.

Заметим, что количество форм языкового описания для модели (1.1) изначально не оговаривается. Фактически это означает то, что слова, являющиеся синонимами по *Лемме 5.1*, могут обозначать понятия с различной степенью абстракции. На практике указанная степень тем больше, чем больше количество ситуаций вида (1.1), относительно которых понятие фигурирует в некоторой фиксированной роли.

Возьмем указанный факт за основу определения меры схожести для ситуаций языкового употребления, порождаемых независимо друг от друга.

Представим тезаурус, формируемый на основе совокупности ситуаций ЕЯ-употребления для известных фактов заданной предметной области, посредством формального контекста:

$$K^{TH} = \left(G^{TH}, M^{TH}, I^{TH}\right). \qquad (5.2)$$

При этом множество объектов G^{TH} составляют символьные пометки, присваиваемые отдельным ситуациям. Множество M^{TH} включает элементы множеств признаков формальных контекстов вида (5.1) всех $g^{TH} \in G^{TH}$. Кроме того, в составе M^{TH} выделяются:

– множество указаний на основы слов, синтаксически подчиненных другим словам в ЕЯ-описаниях ситуаций $g^{TH} \in G^{TH}$. Фактически данное множество, обозначаемое далее как M_6, содержит указания на объекты формальных контекстов вида (5.1), генерируемых для элементов G^{TH};

– множество связей “основа-флексия” для синтаксически зависимого слова, M_7;

– множество сочетаний основ зависимого и главного слова, M_8.

На рис. 5.3 формальный контекст из примера на рис. 5.2 представлен одним ФП для объекта $g^{TH} \in G^{TH}$.

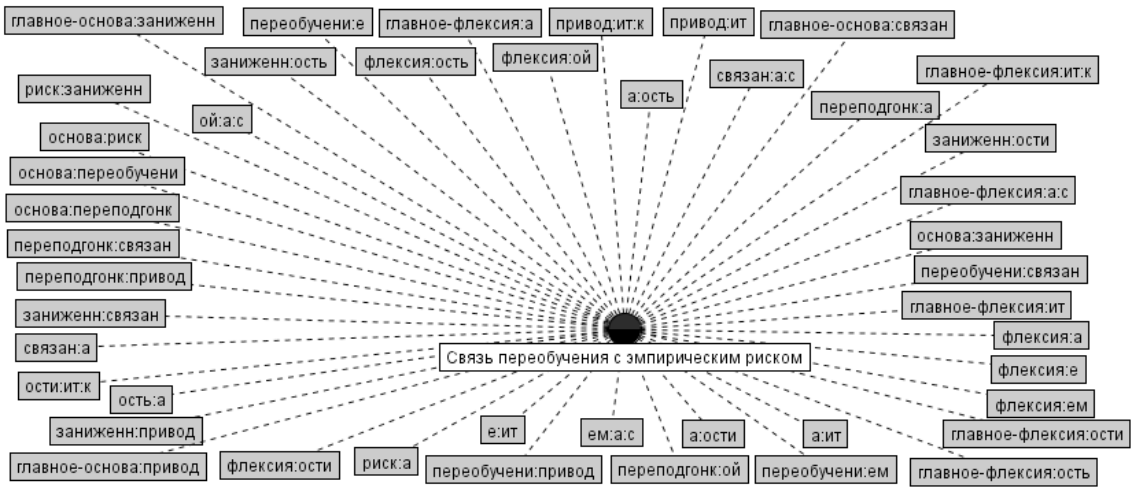


Рис. 5.3. Ситуация ЕЯ-употребления как объект формального контекста тезауруса

Таблица 5.1

Исходные данные для построения тезауруса

№ п/п	1				2	3		4	
Основа	Флективная часть + предлог								
заниженн	ость	ость	ости	ости	–	ость	ости	ость	ость
оценк	–	–	–	–	–	и	и	и	и
эмпирическ	ого	–	ого	–	–	–	–	–	–
риск	а	–	а	–	–	–	–	–	–
средн	–	ей	–	ей	–	–	–	–	–
ошибк	–	и:на	–	и:на	–	–	–	и	и
распознавани	–	–	–	–	–	–	–	я	я
обучающ	–	ей	–	ей	–	–	–	–	–
выборк	–	е	–	е	–	–	–	–	–
переусложнени	ем	ем	е	е	–	–	–	–	–
модел	и	и	и	и	–	–	–	–	–
уменьшени	–	–	–	–	е	–	–	–	–
обобщающ	–	–	–	–	ей	ей	ей	–	–
способност	–	–	–	–	и	и	и	–	–
выбор	–	–	–	–	–	–	–	ом	а
решающ	–	–	–	–	его	–	–	его	его
дерев	–	–	–	–	а	–	–	–	–
правил	–	–	–	–	–	–	–	а	а
алгоритм	–	–	–	–	–	а	а	–	–
переподгонк	–	–	–	–	ой	ой	а	–	–
переобучени	–	–	–	–	–	ем	е	–	–
связан	а:с	а:с	–	–	о:с	а:с	–	а:с	–
вызван	а	а	–	–	–	а	–	–	–
обусловлен	а	а	–	–	о	–	–	–	–
привод	–	–	ит:к	ит:к	–	–	ит:к	–	–
завис	–	–	–	–	–	–	–	–	ит:от

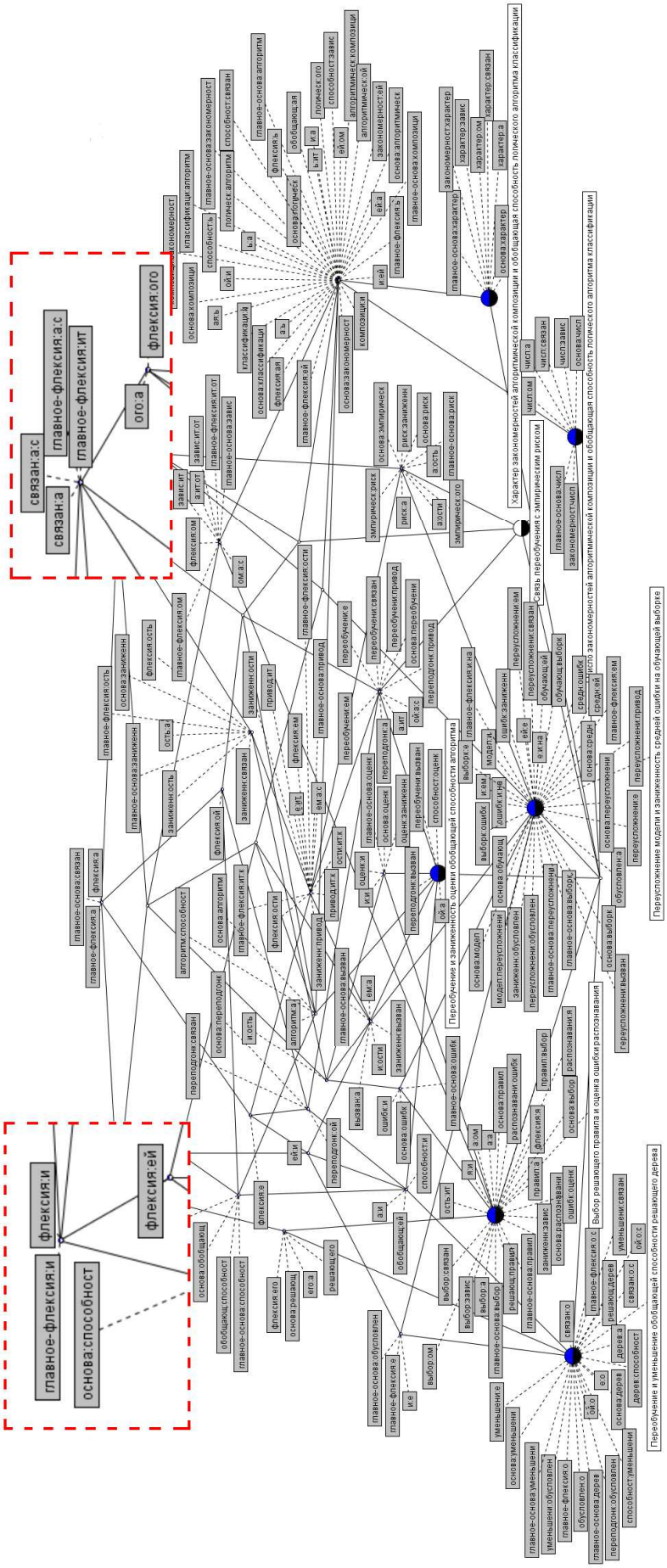


Рис. 5.4. Решетка ФП тезауруса и классы синтаксических отношений

Другие факты этой же предметной области “Математические методы обучения по прецедентам”, использованные для генерации тезауруса, приведены в табл. 5.1. Модель тезауруса в виде решетки формальных понятий представлена на рис. 5.4.

Пусть S_1 – ситуация вида (1.1), соответствующая заведомо корректному (“эталонному”) ЕЯ-описанию некоторого известного факта заданной предметной области. Положим также, что S_2 – анализируемая ситуация, для которой соответствие ситуации S_1 и имеющимся предметным знаниям заранее неизвестно. Обозначим используемые в дальнейших рассуждениях формальные контексты вида (5.1): для ситуации S_1 – как K^E, а для ситуации S_2 – как K^X, где $K^E = \left(G^E, M^E, I^E\right)$ и $K^X = \left(G^X, M^X, I^X\right)$, $I^E \subseteq G^E \times M^E$ и $I^X \subseteq G^X \times M^X$, соответственно. Введем также обозначения для используемых далее символьных констант: p_{fl} – для “флексия:”, p_b - для “основа:”. В соответствии с показанным выше разделением множества признаков формального контекста вида (5.1) будем обозначать соответствующие подмножества в составе M^E и M^X как M_k^E и M_k^X, $k = 1,\ldots,5$. Множество, получаемое объединением множеств M_6, M_7, M_8, M_4^E, M_4^X, M_5^E и M_5^X, обозначим как M^U.

Определение 5.1. Будем считать, что ситуации S_1 и S_2 связаны отношением схожести, если каждому объекту $g^X \in G^X$ соответствует такой объект $g^E \in G^E$, что выполняется одно из следующих условий:

(1) $g^X = g^E$ и любой признак $m^E \in M^E$ объекта g^E будет относиться и к объекту g^X.

(2) $g^X = g^E$, при этом *Условие (1)* не выполняется, но существует объект $g^{TH} \in G^{TH}$, обладающий признаком $m_1^{TH} \in M_6$: $m_1^{TH} = p_b \bullet g^E$ при обязательном выполнении следующих условий:

$\left(\exists\, m_{fl}^E \in M_5^E : m_{fl}^E = p_{fl} \bullet f^E\right) \rightarrow \left(\exists\, m_{17}^{TH} \in M_7 : m_{17}^{TH} = g^E \bullet ":" \bullet f^E\right)$,

при этом $\left(I^E\left(g^E, m_{fl}^E\right) \wedge I^X\left(g^E, m_{fl}^E\right)\right) \rightarrow I^{TH}\left(g^{TH}, m_{17}^{TH}\right)$;

$\left(\exists\, m_{bs}^E \in M_1^E : m_{bs}^E = p_{bs} \bullet b^E\right) \rightarrow \left(\exists\, m_{18}^{TH} \in M_8 : m_{18}^{TH} = g^E \bullet ":" \bullet b^E\right)$,

при этом $I^E\left(g^E, m_{bs}^E\right) \rightarrow I^{TH}\left(g^{TH}, m_{18}^{TH}\right)$;

$\left(\exists\, m_{bs}^X \in M_1^X : m_{bs}^X = p_{bs} \bullet b^X\right) \rightarrow \left(\exists\, m_{28}^{TH} \in M_8 : m_{28}^{TH} = g^E \bullet ":" \bullet b^X\right)$,

при этом $I^X\left(g^E, m_{bs}^X\right) \rightarrow I^{TH}\left(g^{TH}, m_{28}^{TH}\right)$.

Кроме того, для $\forall m^{TH} \in \left(M^{TH} \setminus M^U\right)$ истинно:

$$I^{TH}\left(g^{TH}, m^{TH}\right) \rightarrow \left(I^E\left(g^E, m^{TH}\right) \wedge I^X\left(g^E, m^{TH}\right)\right). \qquad (5.3)$$

В содержательном плане *Условие (2)* настоящего *Определения* описывает случай наличия синонимов среди слов, синтаксически главных по отношению к словам со сходными основами. При этом основы g^X и g^E не омонимичны, поскольку в этом случае было бы нарушено требования разделения ими признаков главного слова.

(3) $g^X \neq g^E$, но существует объект $g^{TH} \in G^{TH}$, обладающий признаками $m_1^{TH} \in M_6$: $m_1^{TH} = p_b \bullet g^E$ и $m_2^{TH} \in M_6$: $m_2^{TH} = p_b \bullet g^X$, при этом для любого признака $m^{TH} \in \left(M^{TH} \setminus M^U\right)$ справедливо:

$$I^{TH}\left(g^{TH}, m^{TH}\right) \rightarrow \left(I^E\left(g^E, m^{TH}\right) \wedge I^X\left(g^X, m^{TH}\right)\right). \qquad (5.4)$$

(4) $g^X \neq g^E$, но существует объект $g_1^{TH} \in G^{TH}$, обладающий признаком $m_1^{TH} \in M_6$: $m_1^{TH} = p_b \bullet g^E$, а для $\forall m^E \in \left(M_4^E \cup M_5^E\right)$ верно:

$$\left(I^{TH}\left(g_1^{TH}, m_1^{TH}\right) \wedge I^E\left(g^E, m^E\right)\right) \rightarrow I^{TH}\left(g_1^{TH}, m^E\right).$$

При этом существуют признаки $m_2^{TH} \in M_6 : m_2^{TH} = p_b \bullet g^{X_1}$ и $m^X \in \left(M_1^X \cup M_2^X \cup M_3^X\right)$, для которых верно:

$$\left(I^{TH}\left(g_1^{TH}, m_2^{TH}\right) \wedge I^X\left(g^X, m^X\right)\right) \rightarrow I^{TH}\left(g_1^{TH}, m^X\right),$$

где $g^{X_1} \neq g^X$, а пара $\left(g^{X_1}, g^E\right)$ отвечает *Условию (3)* настоящего *Определения* при генерации формального контекста вида (5.1) для объекта g_1^{TH}. В то же время существует объект $g_2^{TH} \in G^{TH}$, относительно которого пара $\left(g^X, g^{X_1}\right)$ также будет отвечать *Условию (3)* настоящего *Определения*. Генерируемый при этом формальный контекст вида (5.1) для объекта g_2^{TH} обозначим как. K^{X_1}. По аналогии с K^E и K^X, введенными выше, $K^{X_1} = \left(G^{X_1}, M^{X_1}, I^{X_1}\right)$.

Замечание. Анализ схожести ситуаций S_1 и S_2 включает сравнение последовательностей двух и более соподчиненных слов. Пример: “*средняя ошибка на обучающей выборке*” ⇔ “*эмпирический риск*”. Выполнимость условий *Определения 5.1* здесь анализируется только для главных слов (в примере это “*ошибка*” и “*риск*”). Сами последовательности считаются взаимно заменяемыми, если возможно их построение по формальному контексту (5.2) на наборе признаков с префиксом p_{bs} для одной и той же ситуации языкового употребления. При этом главные слова последовательностей должны быть

одинаково подчинены одному и тому же слову, что проверяется по сочетанию флексий.

Таким образом, *Определение 5.1* учитывает уровень абстракции понятий, обозначаемых словами с основами g^X и g^E, при сходстве их синтаксических ролей, определяемых признаками из множеств M_4^E, M_4^X, M_5^E и M_5^X. При этом само синтаксическое отношение выступает своего рода обобщением ряда семантических отношений. Это подтверждается, в частности, анализом классов ФП в решетке, генерируемой на основе ЕЯ-описаний известных фактов предметной области: отношениям, определяемым сочетаниями флексий, как правило, соответствуют классы более высокого уровня абстракции (в примере на рис. 5.4 эти классы выделены прямоугольниками). Сказанное позволяет в целом провести аналогию между схожестью формальных понятий в рамках одного контекста и схожестью самих формальных контекстов. Этому вопросу посвящен следующий раздел.

5.4. Интерпретация меры схожести формальных понятий для формальных контекстов

Понятие схожести между языковыми контекстами, определяемыми структурами вида (1.1), определяется индуктивно на основе представления о семантическом расстоянии между отдельными лексемами, обсуждавшегося в докладе [82].

Действительно, семантическая схожесть как разновидность семантического расстояния основана на отношении порядка, которое включает родовидовое отношение, отношение синонимии, отношение сочинения и отношение атрибуции между объектами и признаками в формальном контексте. А поскольку только отношение порядка

может быть извлечено из решетки ФП, именно данный вид отношений и должен служить основой схожести между языковыми контекстами.

Согласно данному в [82] определению, полная синонимия между словами с основами $\{g_1, g_2\} \subset G^S$ будет иметь место тогда, когда объекты g_1 и g_2 принадлежат объему одного и того же понятия контекста некоторой ситуации языкового употребления. Фактически именно этот случай и обобщается *Условием (1) Определения 5.1* уже на взаимно различные формальные контексты. Отношение сочинения, как показано в [82], существует между объектами формальных понятий с одинаковым НОСП. Частные случаи такого отношения для объектов из взаимно различных формальных контекстов описывается *Условиями (2)* и *(3) Определения 5.1*.

Более сложные случаи отношения порядка на основе композиции сочинения и родовидового отношения (гипонимии) рекурсивно определяет *Условие (4) Определения 5.1*. Как следует из данного условия, и для взаимно различных формальных контекстов схожесть объектов тем больше, чем более специфичным является их НОСП.

Таким образом, основой меры схожести формальных контекстов должна быть общая информация, разделяемая объектами из разных контекстов, а также специфичность общей информации, вычисляемой по расстоянию от вершины в иерархии контекстов, которая в рассматриваемой нами задаче представляется решеткой для формального контекста вида (5.2).

Обобщая *Определение 5.1*, будем считать, что формальные контексты связаны отношением схожести, если каждому ФП одного контекста можно поставить в соответствие такое ФП второго

контекста, что при этом между формальными понятиями становится возможным установление отношения порядка.

Для введения количественной меры схожести между формальными контекстами рассмотрим обобщенный способ прочтения формул (5.1) и (5.2).

Множество G^{TH} в структуре (5.2) составляют символьные пометки, присваиваемые отдельным контекстам вида (5.1). Объединение множеств M_7 и M_8 в общем случае получает содержательную интерпретацию множества связей между признаками из множества M^{TH}, каждая из которых соответствует некоторой связи объекта и признака конкретного формального контекста $g^{TH} \in G^{TH}$ в представлении (5.1). Таким образом, на основе совокупности структур (5.1) и (5.2) могут быть рекурсивно определены многоуровневые формальные контексты по аналогии с сетями Петри высокого уровня [43], характерный пример которых был фактически рассмотрен нами во второй главе. Мера схожести формальных понятий из контекстов одного уровня рекурсивного вложения определяется аналогично схожести формальных понятий внутри одного контекста. При этом для применения соотношений, описанных в [82], объекты и признаки пары сравниваемых формальных контекстов вида (5.1) должны быть трансформированы в признаки формального контекста вида (5.2), множество типа M_6 для которого содержит указания на объекты обоих формальных контекстов из указанной пары. При установлении степени схожести ситуаций языкового употребления число вышеуказанных уровней рекурсивного вложения равно двум: нижний уровень представлен формальными контекстами сравниваемых ситуаций, верхний – тезаурусом предметной области.

5.5. Семантическая схожесть фраз предметно-ориентированного подмножества естественного языка

Рассмотрим применение модели (5.2) для вычисления меры схожести ситуаций языкового употребления, представляемых формальными контекстами вида (5.1). За основу возьмем предложенную в [82] меру схожести для формальных понятий в пределах одной решетки.

С учетом выполняемого в соответствии с *Определением 5.1* сопоставления объектов формальных контекстов $K^E = \left(G^E, M^E, I^E\right)$ и $K^X = \left(G^X, M^X, I^X\right)$, из которых удалена информация РПЗ, мера схожести ситуаций S_1 и S_2 вычисляется как

$$spc\left(S_1, S_2\right) = \frac{\sum_{k=1}^{n} spc_k}{n}, \qquad (5.5)$$

где $n = \left|G^X\right|$, а spc_k есть мера мера схожести объектов в паре $\left(g_k^X, g^E\right)$. В зависимости от выполнимости условий *Определения 5.1*, значение spc_k:

- равно 1.0, если для пары $\left(g_k^X, g^E\right)$ выполнено *Условие (1)*;
- вычисляется по формуле:

$$-\log_2\left(1 - \frac{D_c}{path_C}\right) \times \frac{\left|B^{LCS}\right|}{\left|B_1 \setminus B^{LCS}\right| + \left|B_2 \setminus B^{LCS}\right| + \left|B^{LCS}\right|}, \qquad (5.6)$$

если для пары $\left(g_k^X, g^E\right)$ выполнено *Условие (2)*, *(3)*, либо *(4)*.

Во втором случае мы имеем дело с гипотетической решеткой ФП (обозначим ее как $\Re^{XE}$), в которой объемы объектных

формальных понятий (формальных понятий с одним объектом в составе объема) есть $\left\{g_k^X\right\}$ и $\left\{g^E\right\}$ (при выполнении *Условия (2)* или *(3)*), либо $\left\{g_k^X\right\}$, $\left\{g^E\right\}$ и $\left\{g^{X_1}\right\}$ (при выполнении *Условия (4)*). Значение D_c равно количеству сравнимых формальных понятий, составляющих цепочку с вершинным ФП решетки $\mathfrak{R}^{XE}$ в качестве максимального ФП и наименьшим общим суперпонятием для объектных формальных понятий решетки $\mathfrak{R}^{XE}$ – в качестве минимального ФП. Множество B^{LCS} есть содержание этого НОСП, а число $path_C$ равно минимальному количеству формальных понятий в цепочке, которой принадлежит вершинное ФП, наименьшее ФП решетки $\mathfrak{R}^{XE}$ и формальное понятие с содержанием B^{LCS}.

В случае выполнения любого из *Условий (2)*, *(3)* или *(4)* значение $D_c = 2$ (доказательство очевидно).

При выполнении *Условия (2)* либо *(3)* число $path_C = 4$, а в множество B^{LCS} войдут признаки $m^{TH} \in \left(M^{TH} \setminus M^U\right)$, для каждого из которых справедливо либо соотношение (5.3) (при выполнении *Условия (2)*), либо соотношение (5.4) (при выполнении Условия (3)). Множества B_1 и B_2 в этом случае определяются следующим образом:

$$B_1 = \left\{m^E : m^E \in \left(M_1^E \cup M_2^E \cup M_3^E\right),\ I^E\left(g^E, m^E\right) = true\right\},$$

$$B_2 = \left\{m^X : m^X \in \left(M_1^X \cup M_2^X \cup M_3^X\right),\ I^X\left(g_k^X, m^X\right) = true\right\}.$$

Доказательство выполнимости *Условия (4)* обычно происходит в несколько итераций. При этом в ходе каждой последующей итерации число признаков, не являющихся общими для g_k^X и g^{X_1}, всегда

меньше, чем в предыдущей. Начальное значение числа $path_C$, равное 4, в ходе каждой итерации увеличивается на 1, а

$$B_1 = \left\{ m^{X_1} : m^{X_1} \in \left(M_1^{X_1} \cup M_2^{X_1} \cup M_3^{X_1} \right),\ I^{X_1}\left(g^{X_1}, m^{X_1} \right) = true \right\},$$

$$B_2 = \left\{ m^{X} : m^{X} \in \left(M_1^{X_1} \cup M_2^{X_1} \cup M_3^{X_1} \right),\ I^{X_1}\left(g_k^{X}, m^{X} \right) = true \right\},$$

где $\left(M_1^{X_1} \cup M_2^{X_1} \cup M_3^{X_1} \right) \subset M^{X_1}$ согласно показанному выше разделению множества признаков формального контекста вида (5.1). Множество B^{LCS} в этом случае есть пересечение множеств B_1 и B_2.

Значения $|B_1|$ и $|B_2|$ в формуле (5.6) будут тем больше, чем большее число слов могут быть синтаксически главными по отношению к каждому из слов для пары $\left(g_k^{X}, g^{E} \right)$. При этом величина $\left| B^{LCS} \right|$ отражает взаимную специфичность понятий, обозначаемых g_k^{X} и g^{E}.

В качестве примера рассмотрим ЕЯ-описание факта наличия связи между переобучением и эмпирическим риском, представленное для ситуации S_1 четырьмя синонимичными простыми распространенными предложениями русского языка.

Предложения 1 и 2: *“Переобучение (=переподгонка) приводит к заниженности эмпирического риска”*. Предложения 3 и 4: *“Заниженность эмпирического риска связана с переподгонкой (=переобучением)”*.

Выполнив синтаксический разбор программой “Cognitive Dwarf”, выделяем основы, флексии и их сочетания. Получаем формальный контекст вида (5.1), представленный решеткой формальных понятий на рис. 5.5.

Рис. 5.5. Ситуация ЕЯ-употребления для "эталонного" описания заданного факта

Теперь предположим, что мы имеем три анализируемых независимых варианта ЕЯ-описания ситуации S_2, причем все три связаны отношением схожести с ситуацией S_1 согласно *Определению 5.1*. Каждый из них описывает тот же факт, что и S_1 – наличие связи между переобучением и эмпирическим риском, причем описание выполнено одним простым распространенным предложением русского языка.

Первый вариант: *“Заниженность средней ошибки на обучающей выборке связана с переобучением”*. Второй вариант: *“Заниженность средней ошибки на обучающей выборке связана с переподгонкой”*. Третий вариант: *“Переобучение приводит к заниженности средней ошибки на обучающей выборке”*.

Как и для ситуации S_1, формальные контексты вида (5.1) здесь строятся на основе результатов синтаксического разбора предложений программой “Cognitive Dwarf”. Полученные решетки формальных понятий представлены на рис. 5.6, 5.7 и 5.8.

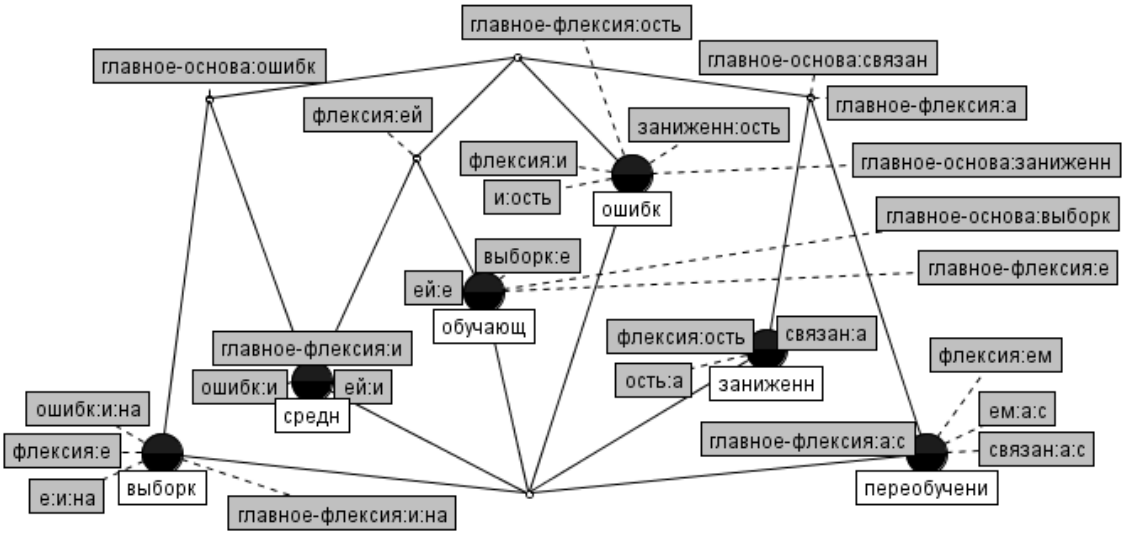


Рис. 5.6. Вариант 1 анализируемого ЕЯ-описания связи переобучения с эмпирическим риском

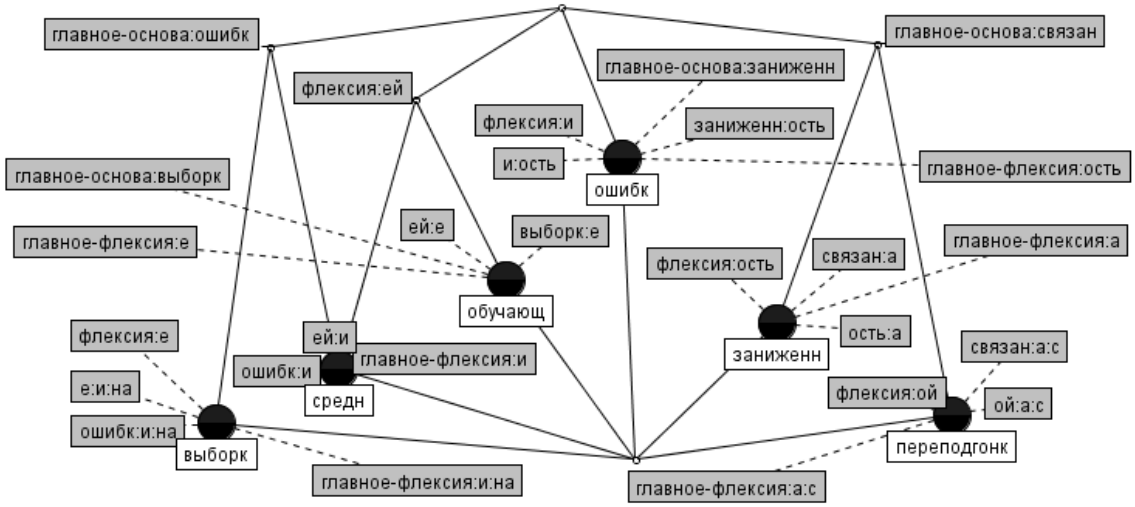


Рис. 5.7. Вариант 2 анализируемого ЕЯ-описания связи переобучения с эмпирическим риском

Таблица 5.2

Сравнение вариантов ЕЯ-описания ситуации S_2

Вариант	$spc(S_1, S_2)$	$\left\|B^{LCS}\right\|$	$\left\|B_1 \setminus B^{LCS}\right\|$	$\left\|B_2 \setminus B^{LCS}\right\|$
1	0,9167	7,7500	0,7500	0,0000
2	0,7917	7,0000	2,0000	0,5000
3	0,8750	7,7500	0,7500	0,7500

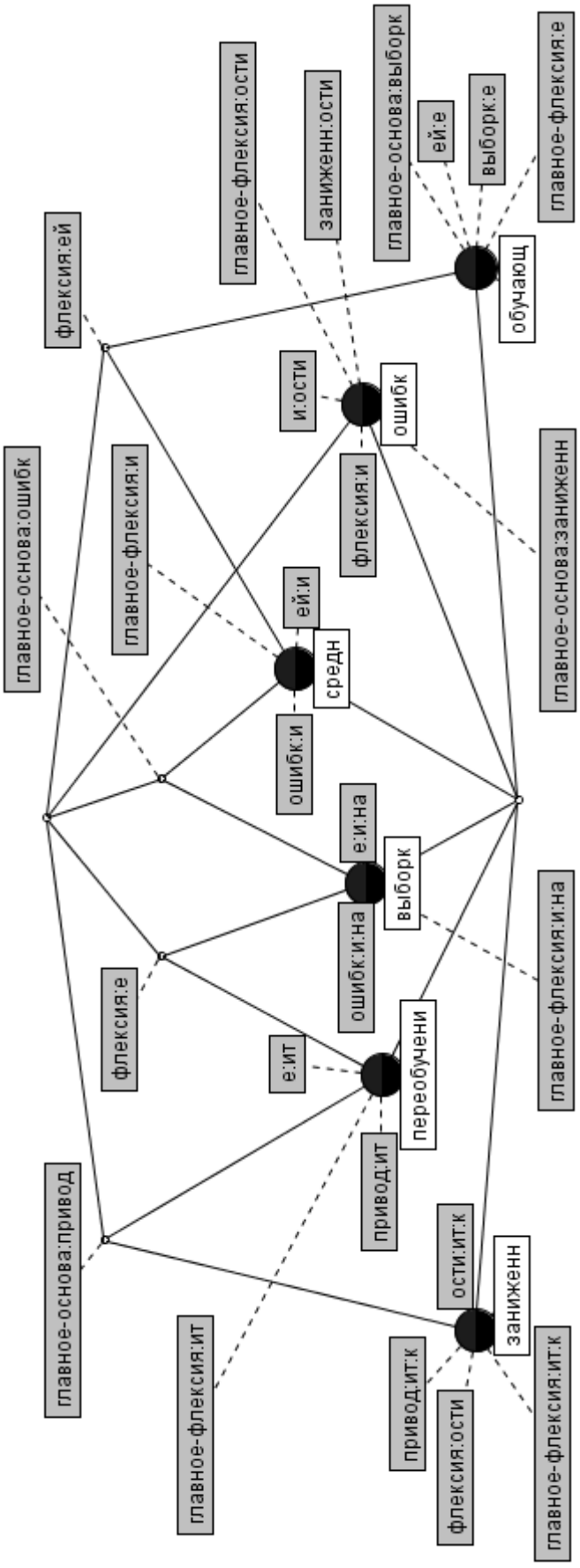


Рис. 5.8. Вариант 3 анализируемого ЕЯ-описания связи переобучения с эмпирическим риском

Как видно из табл. 5.2, наибольшее значение схожести с ситуацией S_1 по формуле (5.5) имеет *Вариант 1* ЕЯ-описания ситуации S_2.

Действительно, для этого варианта в формуле (5.6) мы имеем наибольшее среднее значение $\left|B^{LCS}\right|$ при минимальном среднем значении суммы $\left|B_1 \setminus B^{LCS}\right|$ и $\left|B_2 \setminus B^{LCS}\right|$ по всем парам $\left(g_k^X, g^E\right)$, для которых выполняется *Условие (2), (3),* либо *(4) Определения 5.1*. Причина состоит в том, что признаки объектов формального контекста, соответствующего *Варианту 1*, разделяются большим количеством объектов формального контекста ситуации S_1, чем признаки у объектов формальных контекстов для *Вариантов 2* и *3*. Иными словами, признаки для *Варианта 1* являются более стереотипическими по отношению к формальному контексту ситуации S_1, чем признаки у двух других вариантов.

Немаловажную роль при вычислении меры схожести ситуаций языкового употребления играет также полнота и непротиворечивость ЕЯ-описания предметных знаний при формировании тезауруса. Предложенная модель тезауруса в виде решетки формальных понятий позволяет задействовать, в частности, базис импликаций формального контекста (5.2) для изучения взаимозаменяемости абстрактных слов в синтаксических контекстах существительных предметной лексики (“*связана с переобучением*” ⇔ “*переобучение приводит (к)*”). Соотнесение соответствующих классов ФП решетки тезауруса с уже известными классами семантической эквивалентности в заданном ЕЯ – тема отдельного рассмотрения.

5.6. Сжатие текстовой информации на основе теоретико-решеточного подхода : проблемы и перспективы

В настоящем разделе мы вкратце остановимся на основных вопросах использования модели (5.2) в качестве основы построения текстовых баз данных для заданной предметной области. Сразу отметим, что полная архитектура СУБД на основе теоретико-решеточного подхода не является предметом рассмотрения в настоящей работе и заслуживает отдельного обсуждения.

Во-первых, для организации самой базы данных в рамках любой из известных на сегодняшний день моделей необходимо определиться с набором отношений, непосредственно определяющих данные. В качестве такого набора вполне может выступать совокупность характеристических функций, определяющих смысл текста. Данное определение естественным образом вытекает из формального определения смысла слова, сформулированного в *Главе 3*, и на основе рассуждений, проделанных нами в *Главах 4* и *5* относительно синтаксического контекста имени существительного.

Во-вторых, при использовании смысла как набора атрибутов текста актуальна проблема избыточности данных, в первую очередь вызванная наличием расщепленных предикатных значений. Согласно общеизвестным правилам нормализации отношений [11], связи между главным и зависимым словом в составе РПЗ, а также между РПЗ и его нерасщепленным эквивалентом, должны быть представлены отдельно от связей между участниками ситуаций и самими ситуациями.

Модель (5.2) решает указанную задачу даже если из формальных контекстов вида (5.1), составляющих основу ее формирования, специально не удалена информация расщепленных предикатных значений согласно *Теореме 5.3*: этим конструкциям

будут соответствовать отдельные области в решетке тезауруса. Для выделения РПЗ в отдельную решетку с последующим анализом ее свойств в этом случае может быть полезным алгоритм сегментации решеток, о котором говорилось в докладе [82].

Помимо указанных преимуществ, модель вида (5.2) решает актуальную для нормализации отношений проблему функциональной зависимости неключевых атрибутов от части составного ключа [11]. Применительно к текстовым базам данных указанная зависимость обусловлена как наличием расщепленных предикатных значений в исходных текстах, так и более широким классом синонимического варьирования в рамках стандартных лексических функций. Оперируя критерием полезности решетки, рассмотренным в *Главе 4*, данную проблему в случае без расщепления лексического значения можно решить либо путем замены слова в тексте на исходное слово-аргумент лексической функции, либо путем выбора того значения ЛФ из нескольких возможных, которое максимизирует полезность решетки.

Следует также отметить еще одну качественную особенность моделей вида (5.2), напрямую связанную с репрезентативностью корпуса текстов, составляющего основу формирования предметных знаний. Как было справедливо отмечено в [124], репрезентативность – это такой тип отображения проблемной области в корпус текстов, при котором последний отражает все свойства проблемной области, релевантные для данного лингвистического исследования. Фактически репрезентативность определяется частотой встречаемости в тексте определенных семантических и синтаксических конструкций из фиксируемых моделью (5.2) и, следовательно, может служить своего рода показателем способности корпуса текстов к сжатию посредством теоретико-решеточного представления.

Связывая репрезентативность исходного корпуса текстов и полезность решетки, отметим, что чем выше репрезентативность корпуса, тем большей полезностью будет обладать решетка для контекста (5.2), что означает и более высокую степень сжатия по сравнению с линейным представлением текстов. Первостепенную роль здесь играет информативность комбинации слов в составе каждой из рассматриваемых конструкций [124]. Весовой коэффициент информативности здесь вычисляется на основе взаимной зависимости слов в составе конструкции. Хорошим примером может послужить поточечный коэффициент взаимной зависимости синтаксически главного w_1 и зависимого слова w_2, обсуждавшийся в [82]:

$$depn(w_2, w_1) = \log_2 \frac{frec(w_2, w_1) \cdot N}{frec(w_2) \cdot frec(w_1)},$$

где $frec(w_2, w_1)$ – частота, с которой w_2 встречается в корпусе как непосредственно синтаксически подчиненное слову w_1; $frec(w_2)$ и $frec(w_1)$ – частоты, с которыми встречаются w_2 и w_1 отдельно в корпусе; N – общее количество слов в корпусе.

Сама репрезентативность корпуса является также показателем отражения в текстах определенного жанра.

Так, для деловой и научной прозы, представленной в формальных решетках на рис. 5.1 – 5.8, характерно строгое разграничение семантико-синтаксических контекстов вида (4.1) между существительными относительно предикатных слов в составе указанных последовательностей. Пример (из табл. 5.1): “*заниженн-ость завис-ит:от (связан-а:с)*”, но “*уменьшени-е связан-о:с*”. При этом сжатие текстов на основе модели (5.2) происходит (в первую очередь) за счет тех предикатных слов, которые либо обозначают ситуации, сходные в той или иной мере по составу участников и

характеру выполняемых ими действий, либо (как в приведенном примере) относятся к абстрактной лексике. В целом же способность текстов различных жанров к сжатию является темой отдельного прикладного исследования.

Выводы

Основная сфера применения предложенного в настоящей главе *метода анализа схожести ситуаций языкового употребления* – задачи семантического анализа, для которых заранее неизвестно соответствие сравниваемых текстов тезаурусной информации в силу независимости их порождения.

Следует отметить, что к числу указанных задач относится и интерпретация текста ответа на тестовое задание открытой формы. Как правило, разработчик теста формулирует один или несколько вариантов “правильного” ответа, опираясь на знания о некоторых соотношениях объектов в заданной предметной области. Вместе с тем факт, описываемый “правильным” ответом, не всегда имеет отражение в тезаурусе. Унифицируемое теоретико-решеточное представление сравниваемых высказываний и тезаурусной информации позволяет максимально просто пополнять тезаурус и эффективно использовать имеющуюся в нем информацию при анализе близости текстов.

Предложенная *модель тезауруса* может быть использована в качестве основы построения текстовых баз данных для заданной предметной области. Организация текстовой базы данных на основе решетки формальных понятий позволяет за счет иерархического представления информации уменьшить как размер самой базы данных, так и время поиска в ней.

ЗАКЛЮЧЕНИЕ

Задачи семантического анализа текстов являются одним из наиболее перспективных приложений идей и методов теории анализа формальных понятий. Выявление понятий и их признаков непосредственно из текстов позволяет строить модели различных сторон языкового поведения человека применением исключительно программ синтаксического анализа и специализированного программного обеспечения, реализующего методы АФП. Наиболее значимые из указанных программных средств распространяются свободно в сети Internet. С учетом роста числа сфер приложения АФП и возрастающего интереса к этому направлению анализа данных, сказанное позволяет расширить круг потенциальных потребителей реализуемых моделей и привлечь исследователей, заинтересованных в развитии этих моделей для решения прикладных задач.

В настоящей работе извлечение потенциальных пар “объект-признак” из синтаксического дерева на выходе синтаксического анализатора “Cognitive Dwarf” выполняет специализированный программный модуль, за основу при реализации которого была взята программа “Dwarfprint” непосредственно в составе пакета “Cognitive Dwarf”. Результаты экспериментальных исследований предложенных в работе моделей говорят о перспективности использования стратегии синтаксического разбора на основе наиболее вероятных связей слов совместно с методикой выделения и классификации синтаксических отношений, предложенной и описанной нами в заключительном разделе третьей главы. Качественный анализ решеток, генерируемых на основе множеств синтаксических контекстов, позволяет делать практические выводы как о границах применимости, так и о

направлениях дальнейшего совершенстования используемых стратегий и правил синтаксического анализа.

Основные научные результаты исследования, выполненного авторами и положенного в основу материала настоящей монографии, состоят в следующем:

1. Разработано теоретико-множественное описание процесса установления семантической эквивалентности для флективного языка в рамках синонимического варьирования на уровне абстрактной лексики. За счет введения в рассмотрение конечного множества корректно формализуемых правил синонимических преобразований на основе аппарата стандартных лексических функций становится возможным оценить взаимную близость смыслов высказываний не зависящим от их предметной области способом и с учетом большинства возможных случаев синонимии в языке.

2. На основе теории анализа формальных понятий предложен комплексный подход к решению задачи пополнения лингвистических информационных ресурсов из текстов естественного языка с последующим упорядочиванием знаний. В качестве базового элемента информационного ресурса рассматривается понятие как единство толкования значения слов и их форм. Предложенная модель тезауруса в виде решетки формальных понятий позволяет оперировать данными на семантическом уровне без потери или недопустимого упрощения объектов и их признаков.

3. Предложен подход к распознаванию семантических повторов в сравниваемых по смыслу текстах естественного языка. В отличие от традиционного подхода с использованием словаря лексических функций, с целью применения единого механизма оперирования лингвистическими знаниями с задачей распознавания семантической эквивалентности предложено рассматривать в качестве элементов

повтора не отдельные значения лексических функций, а комбинации таких значений вместе со связывающими их отношениями глубинного синтаксиса – лексические синонимические конструкции.

4. Решена задача построения системы целевых выводов в Δ-грамматике. В отличие от традиционных подходов к формализации преобразований синтаксических структур, с целью нахождения последовательности преобразований с заданными свойствами было предложено исследовать динамику функционирования совокупности правил Δ-грамматики на основе ее информационно-логической модели. Разработанная модель учитывает недетерминированный характер порождения множества помеченных деревьев, а построение целевого вывода сводится к классическим задачам сетей Петри.

5. Предложен подход к построению единого семантического образа на уровне глубинного синтаксиса. В отличие от предлагаемого в теории "Смысл⇔Текст" подхода к построению образов суммарного смысла нескольких фраз, с целью использования единого механизма оперирования лингвистическими знаниями предложено суммировать деревья глубинного синтаксиса, приведенные к виду с одинаковой ЛСК, с применением техники суммирования СемП, но без построения последних.

6. Применительно к ситуациям семантической эквивалентности, основанным на расщеплении лексического значения, разработана методика выявления и обобщения семантического отношения между словами, относительно которых задается расщепленное значение. Ее новизна состоит в сопоставлении лексикографических толкований слова, называющего ситуацию в расщепленном значении, и слова, обозначающего ту же ситуацию и эквивалентного по смыслу расщепленному значению. Поскольку толкование нерасщепленного значения посредством названия семантического отношения ссылается

на толкование слова, обозначающего ситуацию в расщепленном значении, формализация толкований на языке логики предикатов первого порядка позволяет описать точную процедуру выявления семантических отношений и их классификации на основе методов анализа формальных понятий.

7. На основе характеристических функций, которые задаются утверждениями теории лексического значения слова и определяют связь толкуемого слова с другими словами и понятиями, выработаны принципы обобщения независимых вариантов толкований слова. Предложена и исследована модель системы элементов толкования из обобщаемых вариантов последнего. Введением в рассмотроение области, которую образуют элементы толкования заданного лексического значения в решетке формальных понятий, определено ключевое правило обобщения утверждений независимых вариантов теории лексического значения.

8. Решена задача выделения и классификации синтаксических групп на основе описаний ситуаций действительности множествами семантически эквивалентных ЕЯ-фраз. Новизна решения заключается в сравнении символьных последовательностей, составляющих эквивалентные по смыслу предложения, с выделением флексий и основ для последующего анализа взаимного расположения слов и устойчивых словосочетаний в предложениях с разными логическими акцентами относительно одной и той же ситуации. Сказанное актуально при исследовании случаев применения определенных грамматических конструкций в тематическом корпусе текстов.

9. Разработана модель процесса выявления закономерностей сосуществования словоформ в линейном ряду. Данная модель дает возможность автоматически выделить лучший способ выражения нужной мысли в заданном естественном языке, что позволит

минимизировать количество ошибок синтаксического анализа при использовании его как инструмента формирования объектов и признаков в задаче текстовой кластеризации.

10. Предложен метод формирования морфологических классов слов и выделения морфологических зависимостей на основе синтаксических групп в ЕЯ-фразах, эквивалентных по смыслу. Его новизна состоит в использовании посылок и заключений импликаций относительно формального контекста потенциальных синтаксических групп для поиска флексий с признаками главного/зависимого слова. Сами морфологические зависимости выделяются по характеру флексии зависимого слова и соответствуют наиболее вероятным синтаксическим связям относительно множества семантически эквивалентных фраз.

11. Разработан комплексный подход к решению задачи формирования и кластеризации понятий на основе синтаксического контекста существительного русского языка. Новизна решения состоит в использовании зависимости лексической сочетаемости слова от его семантического класса. При этом понятия, обозначающие участников тех или иных ситуаций, выделяются на основе последовательности из предикатного слова и соподчиненных друг другу существительных. Данный метод позволяет путем рассмотрения отношений между произвольным словом последовательности и ее крайним правым членом, обозначающим выявляемое понятие, распознавать частичные смысловые эквивалентности, задаваемые, в частности, генитивными конструкциями.

12. Предложены методы выявления и систематизации конверсивов и расщепленных предикатных значений в составе синтаксических контекстов существительных.

13. Разработана методика выделения из текстов и кластеризации семантических отношений в заданной предметной области. Сказанное актуально как для ролевой идентификации сущностей относительно ситуации при формировании признаков сравниваемых текстов, так и для разработки синтаксических стратегий и правил. При этом качественные оценки формируемых знаний могут быть даны на основе мер схожести решеток по аналогии с мерами схожести для формальных понятий.

14. Предложен метод анализа схожести ситуаций языкового употребления при их независимом порождении с целью описания ситуаций (явлений) средствами заданного естественного языка. Новизной данного метода является использование теоретико-решеточного представления ситуации языкового употребления в качестве информационной единицы тезауруса предметной области. При этом унифицируемое представление сравниваемых высказываний и тезаурусной информации позволяет максимально просто пополнять тезаурус и эффективно использовать имеющуюся в нем информацию при анализе близости текстов. Применение предложенной модели тезауруса в качестве основы построения текстовых баз данных дает возможность за счет иерархического представления информации сократить размер базы и время поиска в ней.

Сфера применения предложенных в работе методов, моделей и алгоритмов не ограничивается задачами установления семантической эквивалентности текстов. Любая интеллектуальная система включает в качестве обязательной составляющей базу знаний. Предложенный в работе подход к формированию тезауруса может служить основой построения онтологий предметных областей для информационно-аналитических систем. При этом используемые в анализе формальных понятий методы концептуальной кластеризации позволяют создавать

такие онтологии параллельно без ограничения природы используемых источников информации.

Материал настоящей монографии основан на публикациях [12-27,33-38,46-65,79,90,91,99-104,107,109-111,113,114,119-123].

Завершая эту работу, следует отметить некоторые наиболее интересные и значимые направления дальнейших изысканий по данному направлению.

Во-первых, отдельного исследования заслуживает включение наречий и прилагательных в состав рассмотренного нами синтаксического контекста существительного. При этом введение в рассмотрение характеристик действий и дополнительных характеристик участников ситуаций даст возможность выделять в анализируемых текстах расщепления с оценочными адъюнктами, а также расщепления на основе синтаксической деривации.

Во-вторых, чрезвычайно интересным является дальнейшее развитие предложенного в работе метода выделения морфологических зависимостей применительно к изменениям в составе основы слова. Здесь следует отметить беглые гласные, чередования гласных и согласных в составе основы, а также вариантные формы основ. В частности, отдельного рассмотрения заслуживает включение в синтаксические контексты вида (4.1) имен числительных, для которых особенно актуально явление чередования в основах. Пример: "триста", "трехсот", "тремстам", "триста", "тремястами", "трехстах". В связи с этим другое немаловажное направление дальнейших исследований – распознавание слов-паронимов в составе синонимичных фраз. Наиболее плодотворные результаты данное исследование даст совместно с количественным изучением вариативности на уровне морфем и лексем русского языка [9].

СПИСОК ЛИТЕРАТУРЫ

1. *Аванесов В. С.* Композиция тестовых заданий. Учебная книга для преподавателей вузов, учителей школ, аспирантов и студентов педвузов [Текст]. М.: Адепт, 1998. 217 с.

2. АОТ: Автоматическая Обработка Текстов [Электронный ресурс] // http://www.aot.ru/ (дата обращения: 19.07.2010).

3. *Апресян Ю. Д.* Избранные труды: в 2 т. Т.I: Лексическая семантика. Синонимические средства языка [Текст]. М.: Шк. “Языки рус. культуры”, 1995. 472 с.

4. *Биркгоф Г.* Теория решеток. Пер. с англ. [Текст]. М.: Наука, 1984. 568 с.

5. *Воронцов К.В.* Обзор современных исследований по проблеме качества обучения алгоритмов [Текст] // Таврический вестник информатики и математики. 2004. № 1. С. 5-24.

6. *Герасимова И.А.* Формальная грамматика и интенсиональная логика [Текст]. М.: Институт философии РАН, 2000. 156 с.

7. *Гладкий А.В.* Грамматики деревьев. I. Опыт формализации преобразований синтаксических структур естественного языка [Текст] / *А.В. Гладкий, И.А. Мельчук* // Сб. "Информационные вопросы семиотики, лингвистики и автоматического перевода", вып. 1. М., 1971. С. 16-41.

8. *Гладкий А.В.* Грамматики деревьев. II. К построению Δ-грамматики для русского языка [Текст] / *А.В. Гладкий, И.А. Мельчук* // Сб. "Информационные вопросы семиотики, лингвистики и автоматического перевода", вып. 4. М., 1974. С. 4-29.

9. *Гусев В.Д.* Алгоритм выявления устойчивых словосочетаний с учетом их вариативности (морфологической и комбинаторной) [Электронный ресурс] / *В.Д. Гусев, Н.В. Саломатина* // Межд. конф.

по компьютерной лингвистике "Диалог-2004". http://www.dialog-21.ru/Archive/2004/Salomatina.htm (дата обращения: 04.08.2010).

10. *Гэри М.* Вычислительные машины и труднорешаемые задачи. Пер. с англ. [Текст] / *М. Гэри, Д. Джонсон*; под ред. А.А. Фридмана. М.: Мир, 1982. 416 с.

11. *Дейт К.Дж.* Введение в системы баз данных. Пер. с англ. [Текст]. М.: Вильямс, 2008. 1327 с.

12. *Емельянов Г. М.* Концептуально-ситуационное моделирование процесса перифразирования высказываний Естественного Языка как обучение на основе прецедентов [Текст] / *Г.М. Емельянов, А.Н. Корнышов, Д.В. Михайлов* // Интеллектуализация обработки информации. Тез. докл. Межд. науч. конференции. Симф.: Крымский научный центр НАН Украины, 2006. С.78-79.

13. *Емельянов Г.М.* Вопросы моделирования семантической связанности для систем понимания текста [Текст] / *Г.М. Емельянов, Д.В. Михайлов* // Распознавание-2001. Сборник материалов 5-й Межд. конф. Курск: Курский гуманитарно-технический институт, Курский государственный технический университет, 2001. Ч. 1. С.56-58.

14. *Емельянов Г.М.* Вопросы моделирования семантической связанности для систем автоматизированного тестирования знаний [Текст] / *Г.М. Емельянов, Д.В. Михайлов* // Всеросс. конф. ММРО-10. М.: АЛЕВ-В, 2001. С.53-56.

15. *Емельянов Г.М.* Вопросы построения механизма суммирования смысла для систем распознавания текстов на естественном языке [Текст] / *Г.М. Емельянов, Д.В. Михайлов* // Методы и средства обработки сложной графической информации. Тез. докл. VI Всероссийской с участием стран СНГ конференции. Нижний Новгород: НИИ прикладной математики и киберентики ННГУ, 2001. С.83-85.

16. *Емельянов Г.М.* Динамическая модель естественного языка в системах пользовательских интерфейсов [Текст] / *Г.М. Емельянов, Д.В. Михайлов, Е.И. Зайцева* // Межд. конф. по компьютерной лингвистике "Диалог-2002". М.: Наука, 2002. Т.2. С. 165-170.

17. *Емельянов Г.М.* Динамическая модель естественного языка в системах пользовательских интерфейсов [Текст] / *Г.М. Емельянов, Е.И. Зайцева, Д.В. Михайлов* // Интеллектуализация обработки информации. Тез. докл. Межд. науч. конф. Симф.: Крымский научный центр НАН Украины, Таврический национальный университет, 2002. С.120-121.

18. *Емельянов Г.М.* К разработке распознающей системы анализа смысловых образов высказываний на естественном языке [Текст] / *Г.М. Емельянов, Е.И. Зайцева, Д.В. Михайлов, Е.П. Курашова* // 6-я Межд. конф. "Распознавание образов и анализ изображений: новые информационные технологии" (РОАИ-6-2002). Тр. конф. Великий Новгород: НовГУ им. Ярослава Мудрого, 2002. Т. 1. С.220-223.

19. *Емельянов Г.М.* Концептуально-ситуационное моделирование процесса перифразирования высказываний Естественного Языка как обучение на основе прецедентов [Текст] / *Г.М. Емельянов, А.Н. Корнышов, Д.В. Михайлов* // Искусственный интеллект. 2006. № 2. С. 72-75.

20. *Емельянов Г.М.* Построение динамической модели естественного языка применительно к разработке языковой базы знаний [Текст] / *Г.М. Емельянов, Е.И. Зайцева, Д.В. Михайлов* // Искусственный интеллект. 2002. № 2. С. 443-446.

21. *Емельянов Г.М.* Построение Модели Управления предикатного слова на основе его лексикографического толкования

[Текст] / *Г.М. Емельянов, Д.В. Михайлов* // Таврический вестник информатики и математики. 2005. № 1. С.35-48.

22. *Емельянов Г.М.* Применение аппарата ограниченных сетей Петри для построения динамической модели естественного языка [Текст] / *Г.М. Емельянов, Е.И. Зайцева, Д.В. Михайлов* // Интеллектуализация обработки информации. Тез. докл. Межд. науч. конф. Симф.: Крымский научный центр НАН Украины, Таврический национальный университет, 2002. С.121-122.

23. *Емельянов Г.М.* Применение реляционной модели представления данных для организации словаря в системе анализа семантической эквивалентности текстов естественного языка [Электронный ресурс] / *Г.М. Емельянов, Д.В. Михайлов, Д.В. Силанов* // Ученые записки Новгородского университета. http://admin.novsu.ac.ru/uni/scpapers.nsf/publications (дата обращения: 06.08.2010).

24. *Емельянов Г.М.* Распознавание сверхфразовых единств при установлении эквивалентности смысловых образов высказываний в общей задаче моделирования языковой деятельности [Текст] / *Г.М. Емельянов, Д.В. Михайлов* // Известия СПбГЭТУ "ЛЭТИ", серия "Информатика, управление и компьютерные технологии", выпуск 1. СПб., 2003. С. 65-73.

25. *Емельянов Г.М.* Синонимические преобразования в задаче анализа эквивалентности смысловых образов высказываний на уровне сверхфразовых единств [Текст] / *Г.М. Емельянов, Д.В. Михайлов, Е.И. Зайцева* // 6-я Международная конференция "Распознавание образов и анализ изображений: новые информационные технологии" (РОАИ-6-2002). Труды конференции. Великий Новгород: НовГУ им. Ярослава Мудрого, 2002. Т. 1. С.215-219.

26. *Емельянов Г.М.* Установление смысловой эквивалентности высказываний: на пути к решению проблемы [Текст] / *Г.М. Емельянов, Д.В. Михайлов* // Искусственный интеллект. 2004. № 2. С. 86-90.

27. *Емельянов Г.М.* Установление смысловой эквивалентности высказываний: на пути к решению проблемы [Текст] / *Г.М. Емельянов, Д.В. Михайлов* // Интеллектуализация обработки информации. Тез. докл. Межд. науч. конф. Симф.: Крымский научный центр НАН Украины, 2004. С.70.

28. *Загоруйко Н.Г.* Прикладные методы анализа данных и знаний [Текст]. Новосибирск: Изд-во Ин-та математики, 1999. 270 с.

29. Искусственный интеллект: в 3-х кн. [Текст] / под ред. Э.В. Попова. М.: Радио и связь, 1990.

30. *Караулов Ю.Н.* Лингвистическое конструирование и тезаурус литературного языка [Текст]. М.: Наука, 1981. 366 с.

31. *Кибрик А.Е.* Очерки по общим и прикладным вопросам языкознания [Текст]. М.: КомКнига, 2005. 332 с.

32. *Кондратов А. М.* Звуки и знаки [Текст]. М.: Знание, 1978. 208 с.

33. *Корнышов А. Н.* Концептуально-ситуационное моделирование высказываний естественного языка в задаче анализа их смысловой эквивалентности [Текст] / *А.Н. Корнышов, Д.В. Михайлов* // Вестник Новгородского гос. ун-та имени Ярослава Мудрого, сер. "Технические науки". 2005. № 34. С. 76-80.

34. *Корнышов А.Н.* Иерархизация системы предикатов семантических отношений [Текст] / *А.Н. Корнышов, Д.В. Михайлов* // Интеллектуализация обработки информации. Тез. докл. Межд. науч. конф. Симф.: Крымский научный центр НАН Украины, 2008. С.130-131.

35. *Корнышов А.Н.* Концептуальный уровень и его использование в задаче моделирования синонимических преобразований высказываний естественного языка [Текст] / *А.Н. Корнышов, Д.В. Михайлов* // Математика в вузе. Материалы XVIII международной научно-методической конф. СПб.: Петербургский государственный университет путей сообщения, 2005. С.118-120.

36. *Корнышов А.Н.* Обучение на основе прецедентов в задаче распознавания смысловой эквивалентности [Текст] / *А.Н. Корнышов, Д.В. Михайлов* // XIII науч. конф. преподавателей, аспирантов и студентов НовГУ. Сборник тезисов докладов. Великий Новгород: НовГУ им. Ярослава Мудрого, 2006. С.136.

37. *Корнышов А.Н.* Предикаты семантических отношений в задаче моделирования системы концептуальных зависимостей в тезаурусе предметной области [Текст] / *А.Н. Корнышов, Д.В. Михайлов* // XIV науч. конф. преподавателей, аспирантов и студентов НовГУ. Сб. тез. докл. Великий Новгород: НовГУ им. Ярослава Мудрого, 2007. С.182-183.

38. *Корнышов А.Н.* Таксономия знаний в задаче распознавания семантических отношений [Текст] / *А.Н. Корнышов, Д.В. Михайлов* // Распознавание-2008. Сб. материалов VIII Межд. конф. Курск: Курский гос. технический ун-т, 2008. Ч. 1. С.183-185.

39. *Котов В.Е.* Сети Петри [Текст]. М.: Наука. Главная редакция физико-математической литературы, 1984. 160 с.

40. *Кубрякова Е.С.* Язык и знание: На пути получения знаний о языке: части речи с когнитив. точки зрения. Роль языка в познании мира [Текст]. М: Изд. "Языки славянской культуры", 2004. 555 с.

41. *Леонтьева Н. Н.* Русский общесемантический словарь (РОСС): структура, наполнение [Текст] // Научно-техническая информация. М.: ВИНИТИ, 1997. 1997. № 12. Сер. 2. С. 5-20.

42. *Леонтьева Н.Н.* О методах смысловой компрессии текста [Электронный ресурс] // X Всероссийская объединенная конференция “Интернет и современное общество” (IMS-2007). http://www.ict.edu.ru/vconf/files/7881.pdf (дата обращения: 03.08.2010).

43. *Ломазова И.А.* Вложенные сети Петри: моделирование и анализ распределенных систем с объектной структурой [Текст]. М.: Научный мир, 2004. 208 с.

44. *Мельников Г. П.* Системная типология языков: Принципы, методы, модели [Текст]. М.: Наука, 2003. 395 с.

45. *Мельчук И. А.* Опыт теории лингвистических моделей “Смысл⇔Текст”: Семантика, синтаксис [Текст]. М.: Шк. “Языки рус. культуры”, 1999. 345 с.

46. *Михайлов Д.В.* Автоматизация накопления знаний о синонимии текстов предметного языка [Текст] / *Д.В. Михайлов, Г.М. Емельянов* // Распознавание-2010. Сб. материалов IX Межд. конф. Курск: Курский гос. технический ун-т, 2010. С.186-188.

47. *Михайлов Д.В.* Вопросы использования предметных и естественных языков в задачах открытого тестирования [Текст] // Великий Новгород – город университетский. Материалы юбилейной научно-практической конференции. Великий Новгород: НовГУ им. Ярослава Мудрого, 2003. С.103-104.

48. *Михайлов Д.В.* Иерархия семантических отношений в задаче построения Модели Управления предикатного слова [Текст] / *Д.В. Михайлов, Г.М. Емельянов* // Распознавание-2005. Сб. материалов 7-й Междунар. конф. Курск: Курский гос. технический ун-т, 2005. С.42-43.

49. *Михайлов Д.В.* Информационное наполнение дерева в задаче исследования динамики функционирования Δ-грамматики [Текст] / *Д.В. Михайлов, Г.М. Емельянов* // Распознавание-2003. Сб. материалов

6-й Межд. конф. Курск: Курский гос технический ун-т, 2003. Ч. 1. С.35-37.

50. *Михайлов Д.В.* Информационно-логическая модель системы правил Δ-грамматики [Текст] / *Д.В. Михайлов, Г.М. Емельянов* // Известия СПбГЭТУ "ЛЭТИ", серия "Информатика, управление и компьютерные технологии", выпуск 3. СПб., 2003. С. 96-102.

51. *Михайлов Д.В.* К вопросу автоматизации пополнения базы данных Лексических Функций в задаче установления смысловой эквивалентности текстов Естественного Языка [Текст] / *Д.В. Михайлов, Г.М. Емельянов* // Вестник Новгородского гос. ун-та имени Ярослава Мудрого, сер. "Технические науки". 2007. № 44. С. 45-49.

52. *Михайлов Д.В.* Кластеризация семантических знаний в задаче распознавания ситуаций смысловой эквивалентности [Текст] / *Д.В. Михайлов, Г.М. Емельянов* // Всеросс. конф. ММРО-13. М.: Макс Пресс, 2007. С.500-503.

53. *Михайлов Д.В.* Модель сортовой системы языка в задаче построения семантического образа высказывания на уровне глубинного синтаксиса [Текст] / *Д.В. Михайлов, Г.М. Емельянов* // Таврический вестник информатики и математики. 2006. № 1. С.79-90.

54. *Михайлов Д.В.* Модель сортовой системы языка в задаче построения семантического образа высказывания на уровне глубинного синтаксиса [Текст] / *Д.В. Михайлов, Г.М. Емельянов* // Интеллектуализация обработки информации. Тез. докл. Межд. науч. конф. Симф.: Крымский научный центр НАН Украины, 2006. С.148-150.

55. *Михайлов Д.В.* Морфология и синтаксис в задаче семантической кластеризации [Текст] / *Д.В. Михайлов, Г.М. Емельянов* // Всеросс. конф. ММРО-14. М.: Макс Пресс, 2009. С.563-566.

56. *Михайлов Д.В.* Пополнение словаря Моделей Управления в задаче анализа семантической эквивалентности текстовых документов [Текст] / *Д.В. Михайлов, Г.М. Емельянов* // Методы и средства обработки сложной графической информации. Тез. докл. VIII Всеросс. науч. конф. Нижний Новгород: ГНУ "НИИ ПМК ННГУ", 2005. С.88-93.

57. *Михайлов Д.В.* Построение модели объекта информационного пространства применительно к исследованию динамики функционирования Δ-грамматик [Текст] / *Д.В. Михайлов, Г.М. Емельянов* // Вестник Новгородского государственного университета имени Ярослава Мудрого, серия "Технические науки". 2004. № 26. С. 131-136.

58. *Михайлов Д.В.* Представление смысла в задаче установления семантической эквивалентности высказываний [Текст] / *Д.В. Михайлов, Г.М. Емельянов* // Вестник Новгородского государственного университета имени Ярослава Мудрого, серия "Технические науки". 2004. № 28. С. 106-110.

59. *Михайлов Д.В.* Применение семантических полей словаря РОСС в задаче построения Модели Управления предикатного слова [Текст] / *Д.В. Михайлов, Г.М. Емельянов* // Всеросс. конф. ММРО-12. М.: Макс Пресс, 2005. С.382-385.

60. *Михайлов Д.В.* Семантическая кластеризация текстов предметных языков (морфология и синтаксис) [Текст] / *Д.В. Михайлов, Г.М. Емельянов* // Компьютерная оптика. 2009. Т.33. № 4. С. 473-480.

61. *Михайлов Д.В.* Формирование и кластеризация знаний о синонимии в рамках стандартных Лексических Функций [Текст] / *Д.В. Михайлов, Г.М. Емельянов* // Сборник научных статей. Великий Новгород: НовГУ им. Ярослава Мудрого, 2009. С. 17-33.

62. *Михайлов Д.В.* Формирование и кластеризация контекстов для существительных русского языка в рамках конверсивных замен [Текст] / *Д.В. Михайлов, Н.А. Степанова, И.И. Юрченко* // Физика и механика материалов. Приложение к научно-теоретическому и прикладному журналу "Вестник Новгородского государственного университета имени Ярослава Мудрого". 2009. № 50. С. 31-34.

63. *Михайлов Д.В.* Формирование и кластеризация понятий в задаче автоматизированного построения тезауруса Предметной Области [Текст] / *Д.В. Михайлов, Г.М. Емельянов* // Распознавание-2008. Сб. материалов VIII Междунар. конференции. Курск: Курский государственный технический университет, 2008. Ч. 2. С.20-22.

64. *Михайлов Д.В.* Формирование и кластеризация понятий на основе множества ситуационных контекстов [Текст] / *Д.В. Михайлов, Г.М. Емельянов, Н.А. Степанова* // Таврический вестник информатики и математики. 2008. № 2. С.79-88.

65. *Михайлов Д.В.* Формирование и кластеризация понятий на основе множества ситуационных контекстов [Текст] / *Д.В. Михайлов, Г.М. Емельянов, Н.А. Степанова* // Интеллектуализация обработки информации. Тезисы докладов Межд. науч. конф. Симф.: Крымский научный центр НАН Украины, 2008. С.168-170.

66. Моделирование языковой деятельности в интеллектуальных системах [Текст] / под ред. А.Е. Кибрика и А.С. Нариньяни. М.: Наука, 1987. 279 с.

67. *Налимов В. В.* Вероятностная модель языка. О соотношении естественных и искусственных языков [Текст]. М.: Наука, 1974. 272 с.

68. *Осипов, Г.С.* Приобретение знаний интеллектуальными системами: Основы теории и технологии [Текст]. М.: Наука, 1997. 112 с.

69. *Павиленис Р.И.* Проблема смысла: Современный логико-философский анализ языка [Текст]. М.: Мысль, 1983. 286 с.

70. *Питерсон Дж.* Теория сетей Петри и моделирование систем. Пер. с англ. [Текст]. М.: Мир, 1984. 298 с.

71. *Позин П.А.* Сравнительный анализ открытого и закрытого ответа на тестовое задание [Текст] / *П.А. Позин, В.Д. Синявский* // Развитие системы тестирования в России. Тез. докл. III Всероссийской научно-методической конференции / Под ред. Л.С. Гребнева. М.: Центр тестирования Министерства образования РФ, 2001. С. 207.

72. *Попов Э.В.* Общение с ЭВМ на естественном языке [Текст]. М.: Наука, 1982. 360 с.

73. *Поспелов Д. А.* Ситуационное управление: теория и практика [Текст]. М.: Наука, 1986. 288 с.

74. Представление знаний в человеко-машинных и робототехнических системах: в 4 т. [Текст] / отчет РГ-18 КНВВТ. М.: ВЦ АН СССР: ВИНИТИ, 1984.

75. Программный пакет синтаксического разбора и машинного перевода [Электронный ресурс] // http://cs.isa.ru:10000/dwarf/ (дата обращения: 18.11.2009).

76. *Рубашкин В.Ш.* Представление и анализ смысла в интеллектуальных системах [Текст]. М.: Наука, 1989. 192 с.

77. *Рыков В.В.* Корпус текстов как семиотическая система и онтология речевой деятельности [Электронный ресурс] // Межд. конф. по компьютерной лингвистике "Диалог-2004". http://www.dialog-21.ru/Archive/2004/Rykov.htm (дата обращения: 28.07.2010).

78. *Севбо И.П.* Структура связного текста и автоматизация реферирования [Текст]. М.:Наука, 1969. 135 с.

79. *Силанов Д.В.* Применение теорий Лексических Значений слов при распознавании ситуаций смысловой эквивалентности [Текст]

/ *Д.В. Силанов, Д.В. Михайлов* // XIV науч. конф. преподавателей, аспирантов и студентов НовГУ. Сборник тезисов докладов. Великий Новгород: НовГУ им. Ярослава Мудрого, 2007. С.183-184.

80. *Смирнова Е. И.* Моделирование структуры состояний сложной системы для задач прогнозирования [Текст] // Искусственный интеллект. 2000. № 2. С. 196-199.

81. *Солганик Г.Я.* Стилистика текста: Учеб. пособие [Текст]. М.: Флинта: Наука, 1997. 253 с.

82. *Степанова Н.А.* Формирование и кластеризация понятий в задаче распознавания образов в пространстве знаний [Текст] / *Н.А. Степанова, Г.М. Емельянов* // Всеросс. конф. ММРО-13. М.: Макс Пресс, 2007. С.206-209.

83. *Тестелец Я. Г.* Введение в общий синтаксис [Текст]. – М.: РГГУ, 2001. 800 с.

84. *Тихомиров И.А.* Интеграция лингвистических и статистических методов поиска в поисковой машине "Exactus" [Электронный ресурс] / *И.А. Тихомиров, И.В. Смирнов* // Международная конференция по компьютерной лингвистике "Диалог-2008". http://www.dialog-21.ru/dialog2008/materials/html/80.htm (дата обращения: 18.11.2009).

85. *Фомичев В.А.* Математические основы представления смысла текстов для разработки лингвистических информационных технологий [Текст] // Информационные технологии. 2002. № 10. С. 16-25; № 11. С. 34-45.

86. *Фомичев В.А.* Формализация проектирования лингвистических процессоров [Текст]. М.: Макс Пресс, 2005. 367 с.

87. *Хомский Н.* Формальные свойства грамматик [Текст] // Кибернетический сб., №2. М., 1961. С. 121-130.

88. *Хомский Н.* Язык и мышление. Пер. с англ. [Текст]. М.: Изд. Моск. ун-та, 1972. 122 с.

89. *Челышкова М.Б.* Теория и практика конструирования педагогических тестов. Учебное пособие [Текст]. М.: Исследовательский центр проблем качества подготовки специалистов, 2001. 410 с.

90. *Юрченко И.И.* Программный комплекс вычисления частотных характеристик глаголов для задачи формирования и кластеризации понятий [Текст] / *И.И. Юрченко, Д.В. Михайлов* // XV науч. конф. преподавателей, аспирантов и студентов НовГУ. Сб. тез. докл. Великий Новгород: НовГУ им. Ярослава Мудрого, 2008. С.245.

91. *Юрченко И.И.* Семантическая кластеризация текстов русского языка [Текст] / *И.И. Юрченко, Д.В. Михайлов* // XVI науч. конф. преподавателей, аспирантов и студентов НовГУ. Сб. тез. докл. Великий Новгород: НовГУ им. Ярослава Мудрого, 2009. Ч. 3. С.34-35.

92. Яндекс. Словари [Электронный ресурс] // http://slovari.yandex.ru (дата обращения: 16.07.2010).

93. *Barbara H. Partee* Genitives, Types and Sorts: The Russian Genitive of Measure [Электронный ресурс] / *Barbara H. Partee, Vladimir Borschev* // http://semanticsarchive.net/Archive/GJlMzYwN/B&P_PossWkshp04.pdf (дата обращения: 12.07.2010).

94. *Beloozerov V.N.* Construction and Use of a Thesaurus in Image Analysis and Processing [Текст] / *V.N. Beloozerov, I.B. Gurevich, D.M. Murashov, Yu.O. Trusova* // Pattern Recognition and Image Analysis. 2003. Vol.13. No. 1. P. 67-69.

95. *Beloozerov V.N.* Representation of the Ontology of an Image Analysis Domain for Optimization of Information Retrieval [Текст] / *V.N.*

Beloozerov, I.B. Gurevich, Yu.O. Trusova // Pattern Recognition and Image Analysis. 2005. Vol. 15. No. 2. P. 358-360.

96. *Beloozerov V.N.* Searching for Solutions in the Image Analysis and Processing Knowledge Base [Текст] / *V.N. Beloozerov, D.M. Murashov, Yu.O. Trusova, D.A. Yanchenko* // Pattern Recognition and Image Analysis. – 2005. Vol.15. No. 2. P. 361-364.

97. *Beloozerov V.N.* Thesaurus for Image Analysis: Basic Version [Текст] / *V.N. Beloozerov, I.B. Gurevich, N.G. Gurevich, D.M. Murashov, Yu.O. Trusova* // Pattern Recognition and Image Analysis. 2003. Vol. 13. No. 4. P. 556-569.

98. *Colantonio S.* Cell Image Analysis Ontology [Текст] / *S. Colantonio, I. Gurevich, M. Martinelli, O. Salvetti, Yu. Trusova* // Pattern Recognition and Image Analysis. 2008. Vol.18. No. 2. P. 332-341.

99. *Emel'yanov G.M.* Clusterization of Semantic Meanings in the Problem of Sense Equivalence Situation Recognition [Текст] / *G.M. Emel'yanov, D.V. Mikhailov* // Pattern Recognition and Image Analysis. 2009. Vol. 19. No. 1. P. 92-102, DOI: 10.1134/S1054661809010179.

100. *Emel'yanov G.M.* Analysis of Semantic Relations in Classification of Sense Images of Statements [Текст] / *G.M. Emel'yanov, D.V. Mikhailov, N.A. Stepanova* // Pattern Recognition and Image Analysis. 2007. Vol. 17. No. 2. P. 258-262.

101. *Emelyanov G.M.* Application of the computer thesaurus for automation of updating of the Government Patterns's dictionary [Текст] / *G.M.Emelyanov, D.V.Mikhailov, N.A.Stepanova* // VI INT. CONGRESS ON MATHEMATICAL MODELING. BOOK OF ABSTRACTS. Nizhny Novgorod: University of Nizhny Novgorod, 2004. P.352.

102. *Emelyanov G.M.* Development of Recognition System of Analysis of Semantic Images of Natural Language Statements [Текст] /

G.M. Emelyanov, E.I. Zaitseva, D.V. Mikhailov, E.P. Kurashova // Pattern Recognition and Image Analysis. 2003. Vol. 13. No.2. P. 251-253.

103. *Emel'yanov G.M.* Filling in the Government-Pattern Dictionary in the Analysis of Equivalence for Sense Images of Statements [Текст] / *G.M. Emel'yanov, D.V. Mikhailov* // Pattern Recognition and Image Analysis. 2007. Vol. 17. No. 2. P. 252-257.

104. *Emelyanov G.M.* Formalization of the word's Lexical Meaning in a problem of recognition of Natural Language's statements's synonymy's situations [Текст] / *G.M. Emelyanov, D.V. Mikhailov* // 8th Int. Conf. "Pattern Recognition and Image Analysis: New Information Technologies" (PRIA-8-2007). Conf. Proc. Yoshkar-Ola: Mari State Technical University, 2007. Vol. 2. P. 253-257.

105. *Emelyanov G.M.* Logical Model Of Hypertext Image Database [Текст] / *G.M. Emelyanov, E.I. Smirnova* // Pattern Recognition and Image Analysis. 1999. Vol. 9. No. 3. P. 458-491.

106. *Emelyanov G.M.* Logical Simulation Algebra of Hypertext Image Database [Текст] / *G.M. Emelyanov, E.I. Smirnova* // Pattern Recognition and Image Analysis. 2000. Vol. 10. No. 1. P. 156-163.

107. *Emelyanov G.M.* Recognition of Superphrase Unities in Texts while Establishing Their Semantic Equivalence [Текст] / *G.M. Emelyanov, D.V. Mikhailov, E.I. Zaitseva* // Pattern Recognition and Image Analysis. 2003. Vol. 13. No. 3. P. 447-451.

108. *Emelyanov G.M.* Semantic Analysis in Computer-Aided Systems of Speech Understanding [Текст] / *G.M. Emelyanov, T.V. Krechetova, E.P. Kurashova* // Pattern Recognition and Image Analysis. 1998. Vol. 8. No. 3. P. 408–410.

109. *Emelyanov G.M.* Semantic relation analysis for classification of meaning pattern of utterances [Текст] / *G.M.Emelyanov, D.V.Mikhailov* // 7th Int. Conf. on Pattern Recognition and Image Analysis: New

Information Technologies (PRIA-7-2004). Conf. Proc. St. Petersburg: SPbETU, 2004. Vol. II. P. 460-461.

110. *Emelyanov G.M.* Semantic Relation Analysis for Classification of the Meaning Patterns of Utterances [Текст] / *G.M. Emelyanov, D.V. Mikhailov, N.A. Stepanova* // Pattern Recognition and Image Analysis. 2005. Vol. 15. No. 2. P. 382-383.

111. *Emelyanov G.M.* Synonymic Transformations in Analysis of Semantic Pattern Equivalence at the Superphrase Unity Level [Текст] / *G.M. Emelyanov, D.V. Mikhailov, E.I. Zaitseva* // Pattern Recognition and Image Analysis. 2003. Vol. 13. No.1. P. 21-23.

112. *Emelyanov G.M.* Tree Grammars in the Problems of Searching for Images by Their Verbal Descriptions [Текст] / *G.M. Emelyanov, T.V. Krechetova, E.P. Kurashova* // Pattern Recognition and Image Analysis. 2000. Vol. 10. No. 4. P. 520–526.

113. *Emelyanov G.M.* Updating of the language knowledge base in the problem of statement's semantic images's equivalence's analysis [Текст] / *G.M.Emelyanov, D.V.Mikhailov* // 7th Int. Conf. on Pattern Recognition and Image Analysis: New Information Technologies (PRIA-7-2004). Conf. Proc. St. Petersburg: SPbETU, 2004. Vol. II. P. 462-465.

114. *Emelyanov G.M.* Updating the Language Knowledge Base in the Problem of Equivalence Analysis of Semantic Images of Statements [Текст] / *G.M. Emelyanov, D.V. Mikhailov* // Pattern Recognition and Image Analysis. 2005. Vol. 15. No. 2. P. 384-386.

115. *Ganter B.* Formal Concept Analysis – Mathematical Foundations [Текст] / *B.Ganter, R.Wille* Berlin: Springer-Verlag, 1999. 284 c.

116. *Gurevich I.B.* An Open General-Purposes Research System for Automating the Development and Application of Information Technologies in the Area of Image Processing, Analysis, and Evaluation [Текст] / *I.B.*

Gurevich, A.V. Khilkov, I.V. Koryabkina, D.M. Murashov, Yu.O. Trusova // Pattern Recognition and Image Analysis. 2006. Vol. 16. No. 4. P. 530-563.

117. *Haan B.J.* IRIS: Hipermedia Services [Текст] / *B.J. Haan, P. Kahn, V.A. Riley, J.H. Coombs, N.K. Meyrowitz* // Communication of the ACM. 1992. Vol. 36. № 1. P. 36-51.

118. *Mel'cuk Igor A.* Explanatory Combinatorial Dictionary of Modern Russian. Semantico-Syntactic Studies of Russian Vocabulary [Текст] / *Igor A. Mel'cuk, Alexander K. Zholkovsky*. Wienna, 1984. 992 с.

119. *Mikhailov D.V.* Application Of The Predicate Word's Lexical Meanings's System For Automation Of Updating Of The Dictionary Of Government Patterns [Текст] / *D.V. Mikhailov, G.M. Emelyanov* // Interactive Systems And Technologies: The Problems of Human-Computer Interaction. Collections of scientific papers. Ulyanovsk: ULSTU, 2005. P. 164-168.

120. *Mikhailov D.V.* Formation and clustering of Russian's nouns's contexts within the frameworks of splintered values [Текст] / *D.V. Mikhailov, G.M. Emelyanov* // 9th Int. Conf. on Pattern Recognition and Image Analysis: New Information Technologies (PRIA-9-2008). Conf. Proc. Nizhni Novgorod: N.I. Lobachevsky State University of Nizhni Novgorod, 2008. Vol. 2. P. 39-42.

121. *Mikhailov D.V.* Forming and clustering of syntactic relations on the bases of Natural Language's using's situations [Текст] / *D.V. Mikhailov, G.M. Emelyanov* // Interactive Systems And Technologies: The Problems of Human-Computer Interaction. Collections of scientific papers. Ulyanovsk: ULSTU, 2009. Vol.III. P. 295-307.

122. *Mikhailov D.V.* Roles's contents of Word's Lexical Meaning's in a problem of recognition of synonymy's situations on the basis of standard Lexical Functions [Текст] / *D.V. Mikhailov, G.M. Emelyanov* // Interactive Systems And Technologies: The Problems of Human-Computer

Interaction. Collections of scientific papers. Ulyanovsk: ULSTU, 2007. P. 159-165.

123. *Mikhailov, D.V.* Formation and clustering of Russian's nouns's contexts within the frameworks of Splintered Values [Текст] / *D.V. Mikhailov, G.M. Emelyanov, N.A. Stepanova* // Pattern Recognition and Image Analysis. 2009. Vol. 19. No. 4. P. 664-672,
DOI: 10.1134/S1054661809040154.

124. *Nadezhda Stepanova* Knowledge acquisition process modeling for question answering systems [Текст] / *Nadezhda Stepanova*, *Gennady Emelyanov* // Когнитивное моделирование в лингвистике: Тр. IX межд. конф. Казань: Казанский гос. ун-т, 2007. С.344-354.

125. *Priss, Uta* Linguistic Applications of Formal Concept Analysis [Текст] // Formal Concept Analysis, Foundations and Applications / In: Ganter; Stumme; Wille (eds.). Berlin: Springer Verlag. LNAI 3626, 2005. P. 149-160.

126. The Concept Explorer [Электронный ресурс] // http://conexp.sourceforge.net (дата обращения: 18.11.2009).

127. ToscanaJ: Welcometo the ToscanaJ Suite [Электронный ресурс] // http://toscanaj.sourceforge.net (дата обращения: 16.07.2010).

128. *Vladimir A. Fomichov* Theory of K-Calculuses as a Powerful and Flexible Mathematical Framework for Building Ontologies and Designing Natural Language Processing Systems [Текст] // 5th International Conference FQAS 2002. Berlin: Springer-Verlag, 2002. P. 183-196.

Приложение

Проргамма формирования модели ситуации языкового употребления на основе семантически эквивалентных фраз. Фрагменты исходного текста на языке Visual Prolog 5.2.

Домены пользовательских типов (файл make_se_situations.inc)

```
rlist=real*

char_list=char*

list_of_char_list=char_list*

list_of_ilist=ilist*

/* Совпадения-несовпадения буквенного состава слова для выделения
   флективной части, описывается структурой word_considering:
   первый объект структуры – порядкавый номер слова (для слов,
   нашедших прообразы со сходной неизменной частью).
   второй объект – совпадающая часть слова.
   третий объект – несовпадающая часть;
   четвертый объект – флаг "рассмотрено". */

word_considering=word_considering(integer,char_list,char_list,string)
sentence_considering=word_considering*
set_of_sentences_considering=sentence_considering*
list_of_set_of_sentences_considering=set_of_sentences_considering*
```

```
/* Вспомогательные структуры для поиска прообразов с минимумом
   несовпадений. */

word_considering_aux=
                word_considering_aux(integer,char_list,char_list,char_list)
word_considering_aux_list=word_considering_aux*

word_considering_aux_incoincident=
                        word_considering_aux_incoincident(integer,integer)
word_considering_aux_incoincident_list=
                                word_considering_aux_incoincident*

/* Часть слова, не меняющаяся
   при синонимическом преобразовании */

invariant_part=invariant_part(integer,char_list)
invariant_part_list=invariant_part*

non_invariant_parts_for_given_invariant=
        non_invariant_parts_for_given_invariant(char_list,list_of_char_list)
non_invariant_parts=non_invariant_parts_for_given_invariant*

/* Описание кластера для заданного буквенного инварианта */

cluster_for_words_with_symbolic_invariant=
        cluster_for_words_with_symbolic_invariant(char_list,
                                                  sentence_considering)
set_of_clusters_for_words_with_symbolic_invariant=
        cluster_for_words_with_symbolic_invariant*
```

Головной модуль программы (файл make_se_situations.pro)

```
include "make_se_situations.inc"
include "make_se_situations.con"
include "hlptopic.con"

predicates
        nondeterm clustering_start(set_of_sentences_considering,
                                   invariant_part_list,
                                   set_of_sentences_considering, ilist).
        nondeterm false_taxons_reveal_with_invariants(
                                      set_of_sentences_considering,
                                      non_invariant_parts,
                                      invariant_part_list,
                                      set_of_sentences_considering,
                                      integer).
        nondeterm efapawwaraftm(set_of_sentences_considering,
                                   non_invariant_parts,
                                   set_of_sentences_considering).
        nondeterm taxons_formation_for_given_pseudophrases_set(
                        set_of_sentences_considering,non_invariant_parts).
        invariants_numbering_for_given_non_invariant_parts(
                        integer,non_invariant_parts,invariant_part_list).
        nondeterm pstnipfic(set_of_sentences_considering,
                              non_invariant_parts, invariant_part_list,
                              set_of_sentences_considering).
        nondeterm invariants_numbers_gather(invariant_part_list,ilist).
        nondeterm orders_of_words_in_sentences(
                                 set_of_sentences_considering, list_of_ilist).
```

```
nondeterm most_significant_indexes_reveal(ilist,list_of_ilist,ilist).
nondeterm words_more_similar_than_differ(char_list,
                                         char_list,char_list).
nondeterm common_prefix(char_list,char_list,char_list).
nondeterm prefix(char_list,char_list,char_list).
nondeterm words_more_similar_than_differ_with_given_search(
                word_considering,sentence_considering,
                sentence_considering,sentence_considering,
                list_of_char_list).
nondeterm words_in_falsetaxon_checking(list_of_char_list,
                                       char_list,char_list).
nondeterm false_taxons_reveal_in_sentence(sentence_considering,
            set_of_clusters_for_words_with_symbolic_invariant,
            sentence_considering).
nondeterm false_taxons_reveal(set_of_sentences_considering,
            set_of_clusters_for_words_with_symbolic_invariant,
            set_of_sentences_considering).
nondeterm false_taxons_merging_with_given(char_list,
            set_of_clusters_for_words_with_symbolic_invariant,
            set_of_sentences_considering,
            set_of_clusters_for_words_with_symbolic_invariant).
nondeterm false_taxons_merging(
              set_of_clusters_for_words_with_symbolic_invariant,
              list_of_set_of_sentences_considering).
nondeterm invariants_for_words_in_false_taxons(integer,
                                    set_of_sentences_considering,
                                    set_of_sentences_considering,
                                    invariant_part_list,integer).
```

```
nondeterm pair_of_phrases_processing(string,integer,integer,
        sentence_considering,sentence_considering,
        sentence_considering,sentence_considering,integer).
nondeterm invariant_part_list_building_for_pair(string,
                sentence_considering,invariant_part_list).
nondeterm phrases_check_by_invariant(string,
        set_of_sentences_considering,invariant_part_list,
        invariant_part_list,set_of_sentences_considering).
nondeterm phrase_check_by_invariant(string,
                invariant_part_list,sentence_considering,
                sentence_considering,invariant_part_list).
nondeterm phrases_transform_invariant_respecting(string,
            set_of_sentences_considering,invariant_part_list,
            set_of_sentences_considering).
nondeterm non_invariant_parts_for_given_invariants(
        invariant_part_list,set_of_sentences_considering,
        non_invariant_parts).
nondeterm non_invariant_parts_for_given_invariant_search(
             char_list,set_of_sentences_considering,
             set_of_sentences_considering,list_of_char_list).
nondeterm nipfgisiss(char_list,sentence_considering,
                list_of_char_list,sentence_considering).
nondeterm false_taxons_transform(integer,
                        list_of_set_of_sentences_considering,
                        set_of_sentences_considering,
                        invariant_part_list,integer).
nondeterm false_taxon_search_for_given_alphabetic_structure(
                                char_list,non_invariant_parts,
                                char_list,char_list).
```

```
nondeterm efpawwaraftm(sentence_considering,
                       non_invariant_parts,sentence_considering).
nondeterm taxon_transforming_respecting_new_invariant(char_list,
                            list_of_char_list,list_of_char_list).
nondeterm search_a_word_with_maximal_affinity_for_given(
                          char_list,sentence_considering,char_list,
                          char_list,char_list,char_list).
nondeterm word_and_phrase_processing(integer,char_list,
                    sentence_considering,word_considering_aux_list,
                    word_considering_aux_list,integer).
word_considering_aux_incoincident_estimate(
                  word_considering_aux_list,word_considering_aux_list,
                  word_considering_aux_incoincident_list).
potential_invariant_taxonomy_estimate(sentence_considering,rlist).
nondeterm pitcfe(sentence_considering).
nondeterm taxon_formation_for_given_invariant(char_list,char_list,
                          set_of_sentences_considering,
                          set_of_sentences_considering,
                          list_of_char_list,list_of_char_list,
                          sentence_considering,sentence_considering).
nondeterm taxon_formation_for_given_pseudophrase(
                    sentence_considering,set_of_sentences_considering,
                    set_of_sentences_considering,non_invariant_parts).
nondeterm wdtnipfic(word_considering,non_invariant_parts,
                             invariant_part_list,word_considering).
nondeterm ptnipfic(sentence_considering,non_invariant_parts,
                          invariant_part_list,sentence_considering).
nondeterm frequency_of_occurence(integer,list_of_ilist,integer).
```

```
nondeterm frequencies_of_occurence(ilist,list_of_ilist,
                                   word_considering_aux_incoincident_list).
nondeterm orders_set_for_most_significant_index(ilist,
                                                list_of_ilist,list_of_ilist).
nondeterm orders_set_for_most_significant_indexes(ilist,integer,
                                   word_considering_aux_incoincident_list,
                                   list_of_ilist,ilist).
nondeterm pair_of_phrases_processing2(sentence_considering,
                                                sentence_considering,
                                                word_considering_aux_list,
                                                word_considering_aux_list).
nondeterm pair_of_phrases_processing1(integer,
                                                word_considering_aux_list,
                                                sentence_considering,
                                                word_considering_aux_list,
                                                word_considering_aux_list).
gather_words_from_word_considering_aux(
                                                word_considering_aux_list,
                                                list_of_char_list).
nondeterm select_by_estimations(
                                   word_considering_aux_incoincident_list,
                                   word_considering_aux_list,
                                   word_considering_aux_list,
                                   word_considering_aux_list,
                                   word_considering_aux_list).
renumbering(integer,word_considering_aux_list,
                                   word_considering_aux_list,
                                   word_considering_aux_list,
                                   word_considering_aux_list,integer).
```

```
nondeterm setting_revealed_conformities(string,
                                        word_considering_aux_list,
                                        word_considering_aux_list,
                                        sentence_considering,
                                        sentence_considering,
                                        sentence_considering,
                                        sentence_considering).
nondeterm setting_revealed_conformity(string,
                                      word_considering_aux,
                                      sentence_considering,
                                      sentence_considering).
nondeterm order_of_words_in_sentence(sentence_considering,ilist).
pcbiaptnit(string,sentence_considering,invariant_part_list,
           sentence_considering,invariant_part_list,
           invariant_part_list).
phrase_check_by_new_invariant(string,invariant_part_list,
                              sentence_considering,sentence_considering).
invariant_to_pseudophrase_transform(invariant_part_list,
                                    sentence_considering).
nondeterm pseudophrase_to_new_invariant_transform(string,
                                                  invariant_part_list,
                                                  sentence_considering,
                                                  invariant_part_list).
search_pseudophrase_for_invariant(string,sentence_considering,
                                  char_list,char_list).
phrase_check_by_new_invariant_word(string,invariant_part,
                                   sentence_considering,
                                   sentence_considering).
```

```
nondeterm word_transform_invariant_respecting(string,
                                              word_considering,
                                              invariant_part_list,
                                              word_considering).
nondeterm phrase_transform_invariant_respecting(string,
                                    sentence_considering,invariant_part_list,
                                    sentence_considering).
nondeterm sort_hoar1(word_considering_aux_list,
                                word_considering_aux_list).
nondeterm sort_hoar2(word_considering_aux_incoincident_list,
                                word_considering_aux_incoincident_list).
nondeterm sort_hoar10(word_considering_aux_incoincident_list,
                                 word_considering_aux_incoincident_list).
nondeterm sort_hoar11(rlist,rlist).
nondeterm partition1(word_considering_aux_list,
                                word_considering_aux,
                                word_considering_aux_list,
                                word_considering_aux_list).
nondeterm partition2(word_considering_aux_incoincident_list,
                                word_considering_aux_incoincident,
                                word_considering_aux_incoincident_list,
                                word_considering_aux_incoincident_list).
nondeterm partition9(rlist,real,rlist,rlist).
list_len(list_of_char_list,integer).
list_len(char_list,integer).
list_len(word_considering_aux_incoincident_list,integer).
list_len(sentence_considering,integer).
list_len(set_of_sentences_considering,integer).
list_len(ilist,integer).
```

```
list_len(list_of_ilist,integer).
append(rlist,rlist,rlist).
append(word_considering_aux_list,
        word_considering_aux_list,
        word_considering_aux_list).
append(word_considering_aux_incoincident_list,
        word_considering_aux_incoincident_list,
        word_considering_aux_incoincident_list).
append(sentence_considering,
        sentence_considering,
        sentence_considering).
append(set_of_sentences_considering,
        set_of_sentences_considering,
        set_of_sentences_considering).
append(invariant_part_list,invariant_part_list,invariant_part_list).
append(char_list,char_list,char_list).
append(non_invariant_parts,
        non_invariant_parts,
        non_invariant_parts).
append(set_of_clusters_for_words_with_symbolic_invariant,
        set_of_clusters_for_words_with_symbolic_invariant,
        set_of_clusters_for_words_with_symbolic_invariant).
nondeterm append(ilist,ilist,ilist).
append(list_of_ilist,list_of_ilist,list_of_ilist).
append(non_predicates_quantity_for_sentences,
        non_predicates_quantity_for_sentences,
        non_predicates_quantity_for_sentences).
nondeterm delete(ilist,list_of_ilist,list_of_ilist).
```

```
nondeterm delete(word_considering,
                 sentence_considering,
                 sentence_considering).
nondeterm member(char,char_list).
nondeterm member(integer,ilist).
nondeterm member(word_considering_aux,
                   word_considering_aux_list).
nondeterm member(char_list,list_of_char_list).
nondeterm member(word_considering,sentence_considering).
nondeterm member(sentence_considering,
                   set_of_sentences_considering).
nondeterm member(ilist,list_of_ilist).
nondeterm member(invariant_part,invariant_part_list).
nondeterm list_set(list_of_char_list,list_of_char_list).
nondeterm list_set(list_of_ilist,list_of_ilist).
nondeterm first_n(word_considering_aux_incoincident_list,
                  integer,
                  word_considering_aux_incoincident_list,
                  word_considering_aux_incoincident_list).
nondeterm unit_sets(word_considering_aux_list,
                    word_considering_aux_list,
                    word_considering_aux_list).
nondeterm unit_sets(sentence_considering,
                    sentence_considering,
                    sentence_considering).
nondeterm unit_sets(ilist,ilist,ilist).
nondeterm unit_sets(list_of_ilist,
                    list_of_ilist,
                    list_of_ilist).
```

```
        nondeterm unit_sets(list_of_char_list,
                            list_of_char_list,
                            list_of_char_list).
        nondeterm put(integer,ilist,ilist).
        nondeterm put(char_list,list_of_char_list,list_of_char_list).
        nondeterm put(word_considering,
                      sentence_considering,
                      sentence_considering).
        nondeterm sub_set(sentence_considering,sentence_considering).
        nondeterm sub_set(ilist,ilist).
        nondeterm min(integer,integer,integer).

clauses
/* Таксономия буквенных инвариантов. Исходные данные:
   SynPhraseList_WordsLists_considering_init – список из списков
      структур типа word_considering для исходного СЭ-множества.
      Неизменная часть каждого слова представлена пустым списком.
   Результаты:
   NumberedInvariantParts – список нумерованных описаний
      буквенного состава тех частей слов, которые не меняются при
      синонимическом перифразировании;
   SynPhraseListTr – список, получаемый из исходного списка
      SynPhraseList_WordsLists_considering_init путем выделения
      неизменяемых и флективных частей слов с учетом найденных
      буквенных инвариантов;
   IndexesForSearch – выявленное множество индексов для буквенных
      инвариантов с наибольшей совокупной частотой встречаемости в
      анализируемых ЕЯ-фразах. */
```

```
clustering_start(SynPhraseList_WordsLists_considering_init,
                 NumberedInvariantParts,
                 SynPhraseListTr,
                 IndexesForSearch): –
    false_taxons_reveal_with_invariants(
                      SynPhraseList_WordsLists_considering_init,
                      FalseTaxonsReprRes,
                      InvarsForFalseTaxonsRes,
                      NotInFalseTaxons1,
                      Next_Counter_of_coincidents),
    efapawwaraftm(NotInFalseTaxons1,
                  FalseTaxonsReprRes,
                  NotInFalseTaxons),
    taxons_formation_for_given_pseudophrases_set(NotInFalseTaxons,
                                                 InvariantParts),
    invariants_numbering_for_given_non_invariant_parts(
                                       Next_Counter_of_coincidents,
                                       InvariantParts,
                                       InvarsForOthers),
    append(InvarsForFalseTaxonsRes,
           InvarsForOthers,
           NumberedInvariantParts),
    append(FalseTaxonsReprRes,
          InvariantParts,
          InvariantPartsWithEndings),
    pstnipfic(SynPhraseList_WordsLists_considering_init,
           InvariantPartsWithEndings,
           NumberedInvariantParts,
           SynPhraseListTr),
```

```
        invariants_numbers_gather(NumberedInvariantParts,
                                  RevealedIndexes),
        orders_of_words_in_sentences(SynPhraseListTr,IndexSequences),

        most_significant_indexes_reveal(RevealedIndexes,
                                        IndexSequences,
                                        IndexesForSearch).

/* Разделение ситуации СЭ – НАЧАЛО */

/* Поиск в предложении слов, для которых буквенный состав
   имеет с заданным словом больше сходств,
   чем различий и которые могут образовать ложные таксоны.

   Пример : "метро" (трансп.) - "метр" (ед.изм.) – НАЧАЛО */

   words_more_similar_than_differ(Symbols1, Symbols2,
                                  Conterminous_part): –
        common_prefix(Symbols1,Symbols2,Conterminous_part),
        prefix(Conterminous_part,Symbols1,Incoincident_part1),
        prefix(Conterminous_part,Symbols2,Incoincident_part2),
        list_len(Conterminous_part,Conterminous_part_len),
        list_len(Incoincident_part1,Incoincident_part1_len),
        list_len(Incoincident_part2,Incoincident_part2_len),
        Conterminous_part_len>=Incoincident_part1_len,
        Conterminous_part_len>=Incoincident_part2_len.

   words_more_similar_than_differ_with_given_search(_,[ ],[ ],[ ],[ ]).
```

```
words_more_similar_than_differ_with_given_search(
                word_considering(0,[ ],Symbols1,"false"),
                [word_considering(0,[ ],Symbols2,"false")|InitSentence],
                [word_considering(0,[ ],Symbols2,"false")|FalseTaxon],
                Others,
                [Conterminous_part|Conterminous_parts]): –
    words_more_similar_than_differ(Symbols1,Symbols2,
                                   Conterminous_part),
    words_more_similar_than_differ_with_given_search(
                                word_considering(0,[],Symbols1,"false"),
                                InitSentence, FalseTaxon, Others,
                                Conterminous_parts).

words_more_similar_than_differ_with_given_search(
                word_considering(0,[ ],Symbols1,"false"),
                [word_considering(0,[ ],Symbols2,"false")|InitSentence],
                FalseTaxon,
                [word_considering(0,[ ],Symbols2,"false")|Others],
                Conterminous_parts): –
    not(words_more_similar_than_differ(Symbols1,Symbols2,_)),
    words_more_similar_than_differ_with_given_search(
                                word_considering(0,[ ],Symbols1,"false"),
                                InitSentence,
                                FalseTaxon,
                                Others,
                                Conterminous_parts).

words_in_falsetaxon_checking([ ],Invariant,Invariant).
```

```
words_in_falsetaxon_checking([Symbols1|Conterminous_parts],
                             Symbols2, Invariant): –
    words_more_similar_than_differ(Symbols1,Symbols2,
                                   Conterminous_part),
    words_in_falsetaxon_checking(Conterminous_parts,
                                 Conterminous_part,Invariant).

false_taxons_reveal_in_sentence([ ],[ ],[ ]).

false_taxons_reveal_in_sentence([Word|Sentence],
                    [cluster_for_words_with_symbolic_invariant(
                                 Invariant,
                                 [Word|FalseTaxon])|
                    FalseTaxons],Others): –
    words_more_similar_than_differ_with_given_search(Word,
                                 Sentence, FalseTaxon,
                                 NotInFalseTaxon,
                                 Conterminous_parts),
    list_len(FalseTaxon,FalseTaxonLen),
    FalseTaxonLen>=1,
    Word=word_considering(_,_,Symbols,_),
    words_in_falsetaxon_checking(Conterminous_parts,
                                 Symbols,Invariant),
    false_taxons_reveal_in_sentence(NotInFalseTaxon,
                                 FalseTaxons,Others).

false_taxons_reveal_in_sentence([Word|Sentence],FalseTaxons,
                                [Word|Others]): –
```

```
    words_more_similar_than_differ_with_given_search(
                                                Word,Sentence,[ ],
                                                NotInFalseTaxon,[ ]),
    false_taxons_reveal_in_sentence(NotInFalseTaxon,
                                    FalseTaxons,Others).

false_taxons_reveal([ ],[ ],[ ]).

false_taxons_reveal([Sentence|Sentences],
                    FalseTaxons,
                    [NotInFalseTaxonsForSentence|NotInFalseTaxons]): –
    false_taxons_reveal_in_sentence(Sentence,
                                    FalseTaxonsForSentence,
                                    NotInFalseTaxonsForSentence),
    false_taxons_reveal(Sentences,FalseTaxons1,NotInFalseTaxons),
    append(FalseTaxonsForSentence,FalseTaxons1,FalseTaxons).

false_taxons_merging_with_given(_,[ ],[ ],[ ]).

false_taxons_merging_with_given(Invariant1,
                                [cluster_for_words_with_symbolic_invariant(
                                    Invariant2,FalseTaxon)|FalseTaxons],
                                [FalseTaxon|FalseTaxonsForGiven],
                                OthersFalseTaxons): –
    words_more_similar_than_differ(Invariant1,Invariant2,_),
    false_taxons_merging_with_given(Invariant1,FalseTaxons,
                                    FalseTaxonsForGiven,
                                    OthersFalseTaxons).
```

```
false_taxons_merging_with_given(Invariant1,
                [cluster_for_words_with_symbolic_invariant(Invariant2,
                                                FalseTaxon)|FalseTaxons],
                FalseTaxonsForGiven,
                [cluster_for_words_with_symbolic_invariant(Invariant2,
                                                    FalseTaxon)|
                                                    OthersFalseTaxons]): –
    not(words_more_similar_than_differ(Invariant1,Invariant2,_)),
    false_taxons_merging_with_given(Invariant1,
                                    FalseTaxons,
                                    FalseTaxonsForGiven,
                                    OthersFalseTaxons).

false_taxons_merging([ ],[ ]).

false_taxons_merging([cluster_for_words_with_symbolic_invariant(
                                Invariant,FalseTaxon)|FalseTaxons],
                        [[FalseTaxon|FalseTaxonsForGiven]|Res]): –
    false_taxons_merging_with_given(Invariant,FalseTaxons,
                                    FalseTaxonsForGiven,
                                    OthersFalseTaxons),
    false_taxons_merging(OthersFalseTaxons,Res).

invariants_for_words_in_false_taxons(Curr_Counter_of_coincidents,
                                [Ph1,Ph2|Rest_of_FalseTaxonReprInit],
                                FalseTaxonReprRes,
                                InvarRes,
                                Next_Counter_of_coincidents): –
```

```
    list_len(Ph1,Ph1_LEN),
    pair_of_phrases_processing("true",
                               Curr_Counter_of_coincidents,
                               Ph1_LEN,Ph1,Ph2,
                               Ph1_for_test_new,
                               Ph2_for_test_new,
                               Next_Counter_of_coincidents),
    invariant_part_list_building_for_pair("true",
                                          Ph2_for_test_new,Invar),
    phrases_check_by_invariant("true",
                        Rest_of_FalseTaxonReprInit,
                        Invar,
                        InvarRes,
                        SynPhraseList_WordsLists_considering_new),
    phrases_transform_invariant_respecting("true",
                       [Ph1_for_test_new,
                        Ph2_for_test_new|
                        SynPhraseList_WordsLists_considering_new],
                       InvarRes,
                       FalseTaxonReprRes).

non_invariant_parts_for_given_invariants([ ],_,[ ]).

non_invariant_parts_for_given_invariants([invariant_part(_,Invariant)|
                                          Invariant_parts],
                    SentencesReprInit,
                    [non_invariant_parts_for_given_invariant(Invariant,
                          TaxonReprRes)|
                          InvariantPartsWithNonInvariants]): –
```

```
    non_invariant_parts_for_given_invariant_search(Invariant,
                                                   SentencesReprInit,
                                                   SentencesReprNext,
                                                   TaxonReprRes),
    non_invariant_parts_for_given_invariants(Invariant_parts,
                                             SentencesReprNext,
                                             InvariantPartsWithNonInvariants).

non_invariant_parts_for_given_invariant_search(_,[ ], [ ], [ ]).

non_invariant_parts_for_given_invariant_search(Invariant,
                                               [SentenceReprInit|SentencesReprInit],
                                               [SentenceReprNext|SentencesReprNext],
                                               Incoincidents): –
    nipfgisiss(Invariant,
               SentenceReprInit,
               Incoincidents1,
               SentenceReprNext),
    non_invariant_parts_for_given_invariant_search(Invariant,
                                                   SentencesReprInit,
                                                   SentencesReprNext,
                                                   Incoincidents2),
    unit_sets(Incoincidents1,Incoincidents2,Incoincidents).

/* Название “nipfgisiss” есть сокращение от “non invariant parts for
   given invariant search in single sentence” (англ.). */

nipfgisiss(_,[ ], [ ], [ ]).
```

```
nipfgisiss(Invariant,[word_considering(_,Invariant,
                                       Incoincident_part,_)|
                                  SentenceRepr],
           Incoincidents,
           SentenceReprRes): –
    nipfgisiss(Invariant,SentenceRepr,Incoincidents1,SentenceReprRes),
    put(Incoincident_part,Incoincidents1,Incoincidents).

nipfgisiss(Invariant,
          [word_considering(Label,Invariant1,Incoincident_part,Flag)|
           SentenceRepr], Incoincidents,
          [word_considering(Label,Invariant1,Incoincident_part,Flag)|
           SentenceReprRes]): –
    not(Invariant=Invariant1),
    nipfgisiss(Invariant, SentenceRepr, Incoincidents, SentenceReprRes).

false_taxons_transform(Res_Counter_of_coincidents,
                       [ ], [ ], [ ],
                       Res_Counter_of_coincidents).

false_taxons_transform(Curr_Counter_of_coincidents,
                       [FalseTaxon|FalseTaxons],
                       FalseTaxonsReprRes,
                       InvarsRes,Res_Counter_of_coincidents): –
    invariants_for_words_in_false_taxons(
                                  Curr_Counter_of_coincidents,
                                  FalseTaxon, FalseTaxonReprRes,
                                  Invar,
                                  Next_Counter_of_coincidents),
```

```
    false_taxons_transform(Next_Counter_of_coincidents,
                           FalseTaxons, FalseTaxonsReprRes1,
                           InvarsRes1, Res_Counter_of_coincidents),
    append(FalseTaxonReprRes,
           FalseTaxonsReprRes1,
           FalseTaxonsReprRes),
    append(Invar,InvarsRes1,InvarsRes).

false_taxons_reveal_with_invariants(Sentences,
                                    InvariantPartsWithNonInvariants,
                                    InvarsRes, NotInFalseTaxons,
                                    Next_Counter_of_coincidents): –
    false_taxons_reveal(Sentences,FalseTaxons,NotInFalseTaxons),
    false_taxons_merging(FalseTaxons,FalseTaxonsMerged),
    false_taxons_transform(1,FalseTaxonsMerged,
                           FalseTaxonsReprRes, InvarsRes,
                           Next_Counter_of_coincidents),
    non_invariant_parts_for_given_invariants(InvarsRes,
                                    FalseTaxonsReprRes,
                                    InvariantPartsWithNonInvariants).

/* Исключаем из всех предложений слова, которые попали в ложные
   таксоны – НАЧАЛО */

false_taxon_search_for_given_alphabetic_structure(AllSymbols,
                    [non_invariant_parts_for_given_invariant(Invariant,
                                                TaxonReprRes)|
                     _InvariantPartsWithNonInvariants],
                    Invariant, Incoincident_part): –
```

```
    member(Incoincident_part,TaxonReprRes),
    append(Invariant,Incoincident_part,AllSymbols).

false_taxon_search_for_given_alphabetic_structure(AllSymbols,
                                                  [_|InvariantPartsWithNonInvariants],
                                                  Invariant,
                                                  Incoincident_part): –
    false_taxon_search_for_given_alphabetic_structure(AllSymbols,
                                                      InvariantPartsWithNonInvariants,
                                                      Invariant,
                                                      Incoincident_part).

/* Название “efpawwaraftm” есть сокращение от “exclude from phrase
   any words which are recognized as false taxons members” (англ.). */

efpawwaraftm([ ],_,[ ]).

efpawwaraftm([word_considering(_,Invariant,Incoincident_part,_)|
                SentenceRepr],
             InvariantPartsWithNonInvariants,
             SentenceReprRes): –
    append(Invariant,Incoincident_part,AllSymbols),
    false_taxon_search_for_given_alphabetic_structure(AllSymbols,
                                                      InvariantPartsWithNonInvariants,
                                                      _NewInvariant,
                                                      _New_Incoincident_part),
    efpawwaraftm(SentenceRepr, InvariantPartsWithNonInvariants,
                 SentenceReprRes).
```

```
efpawwaraftm([word_considering(Label,
                               Invariant,Incoincident_part,Flag)|
              SentenceRepr],
              InvariantPartsWithNonInvariants,
              [word_considering(Label, Invariant,
                               Incoincident_part,Flag)|
               SentenceReprRes]): –
    append(Invariant,Incoincident_part,AllSymbols),
    not(false_taxon_search_for_given_alphabetic_structure(
                               AllSymbols,
                               InvariantPartsWithNonInvariants,
                               _,
                               _)),
    efpawwaraftm(SentenceRepr,
                 InvariantPartsWithNonInvariants,
                 SentenceReprRes).

/* Название “efapawwaraftm” есть сокращение от “exclude from all
   phrases any words which are recognized as false taxons members”
   (англ.). */

efapawwaraftm([ ],_,[ ]).

efapawwaraftm([SentenceRepr|SentencesRepr],
              InvariantPartsWithNonInvariants,
              [SentenceReprNew|SentencesReprNew]): –
    efpawwaraftm(SentenceRepr,
                 InvariantPartsWithNonInvariants,
                 SentenceReprNew),
```

```
    efapawwaraftm(SentencesRepr,
                  InvariantPartsWithNonInvariants,
                  SentencesReprNew).

/* Исключаем из всех предложений слова, которые попали в ложные
   таксоны – КОНЕЦ */

/* Поиск в предложении слов, для которых буквенный состав имеет
   больше сходств, чем различий – КОНЕЦ */

/* Преобразование таксона с учетом вновь выявленного инварианта –
   НАЧАЛО */

   taxon_transforming_respecting_new_invariant(_,[ ],[ ]).

   taxon_transforming_respecting_new_invariant(Invariant_new,
                                               [Invariant|InvariantLists],
                                               Others_res): –
      prefix(Invariant_new,
             Invariant,
             Rest),
      taxon_transforming_respecting_new_invariant(Invariant_new,
                                                  InvariantLists,
                                                  Others_res1),
      put(Rest,Others_res1,Others_res).

/* Преобразование таксона с учетом вновь выявленного инварианта –
   КОНЕЦ */
```

```
/* Поиск в предложении слова, максимально близкого заданному по
   буквенному составу */

  search_a_word_with_maximal_affinity_for_given(Incoincident_part1,
                                                Sentence, Conterminous_part,
                                                New_Incoincident_part1,
                                                New_Incoincident_part2,
                                                Incoincident_part2): –
      word_and_phrase_processing(0, Incoincident_part1,
                                    Sentence, Aux1_unsorted,
                                    Aux2_unsorted, _),
      sort_hoar1(Aux1_unsorted,Aux1),
      sort_hoar1(Aux2_unsorted,Aux2),
      word_considering_aux_incoincident_estimate(Aux1, Aux2,
                                                 Aux3_unsorted),
      sort_hoar2(Aux3_unsorted,Aux3),
      Aux3=[word_considering_aux_incoincident(Counter,_)|_],
      member(word_considering_aux(Counter, Conterminous_part,
                                  New_Incoincident_part1,
                                  Incoincident_part1), Aux1),
      member(word_considering_aux(Counter, Conterminous_part,
                                  New_Incoincident_part2,
                                  Incoincident_part2), Aux2).

/* Оценка качества таксономии потенциальных инвариантов с
   вычислением оценки - НАЧАЛО */

  potential_invariant_taxonomy_estimate([ ],[ ]).
```

```
potential_invariant_taxonomy_estimate(
                                [word_considering(_,Conterminous_part,
                                Incoincident_part,_)|ForEstim],
                                [Estimation|Estimations]): –
    list_len(Conterminous_part,Conterminous_part_len),
    list_len(Incoincident_part,Incoincident_part_len),
    Estimation=Conterminous_part_len/
                    (Conterminous_part_len+Incoincident_part_len),
    potential_invariant_taxonomy_estimate(ForEstim,Estimations).

pitcfe(ForEstim): –
    potential_invariant_taxonomy_estimate(ForEstim,
                                                Estimations_unsorted),
    sort_hoar11(Estimations_unsorted,Estimations),
    Estimations=[Min|_Others],
    Min>0.5.
```

/* Оценка качества таксономии потенциальных инвариантов с вычислением оценки – КОНЕЦ */

/* Формирование таксона для заданного инварианта */

```
taxon_formation_for_given_invariant(InvariantPartRes,
                                    InvariantPartRes,
                                    [ ], [ ],
                                    InvariantsListRes,
                                    InvariantsListRes,
                                    ForEstim, ForEstim).
```

```
taxon_formation_for_given_invariant(InvariantPartCurr,
                                    InvariantPartRes,
                                    [PseudoPhrase|PseudoPhraseSet],
                                    [NewPseudoPhrase|NewPseudoPhraseSet],
                                    InvariantsListCurr,
                                    InvariantsListRes,
                                    ForEstimCurr,
                                    ForEstimRes): –
    search_a_word_with_maximal_affinity_for_given(
                                                  InvariantPartCurr,
                                                  PseudoPhrase,
                                                  InvariantPartNext,
                                                  New_Incoincident_part1,
                                                  New_Incoincident_part2,
                                                  Incoincident_part2),
    put(Incoincident_part2,InvariantsListCurr,InvariantsListNext),
    put(word_considering(0,InvariantPartNext,
                                 New_Incoincident_part1,"false"),
        ForEstimCurr,
        ForEstimNext1),
    put(word_considering(0,InvariantPartNext,
                                 New_Incoincident_part2,"false"),
        ForEstimNext1,
        ForEstimNext),
    pitcfe(ForEstimNext),!,
    delete(word_considering(0,[ ], Incoincident_part2,"false"),
           PseudoPhrase,
           NewPseudoPhrase),
```

```
    taxon_formation_for_given_invariant(InvariantPartNext,
                                        InvariantPartRes,
                                        PseudoPhraseSet,
                                        NewPseudoPhraseSet,
                                        InvariantsListNext,
                                        InvariantsListRes,
                                        ForEstimNext,
                                        ForEstimRes).

taxon_formation_for_given_invariant(InvariantPartCurr,
                                    InvariantPartRes,
                                    [PseudoPhrase|PseudoPhraseSet],
                                    [PseudoPhrase|NewPseudoPhraseSet],
                                    InvariantsListCurr, InvariantsListRes,
                                    ForEstimCurr, ForEstimRes): –
    taxon_formation_for_given_invariant(InvariantPartCurr,
                                        InvariantPartRes,
                                        PseudoPhraseSet,
                                        NewPseudoPhraseSet,
                                        InvariantsListCurr,
                                        InvariantsListRes,
                                        ForEstimCurr,
                                        ForEstimRes).

/* Формирование таксона для заданного множества буквенных
   инвариантов */

taxon_formation_for_given_pseudophrase([ ], PseudoPhraseSetRes,
                                       PseudoPhraseSetRes, [ ]).
```

```
taxon_formation_for_given_pseudophrase(
                [word_considering(0,[], Incoincident_part,"false")|
                  PseudoPhrase],
                PseudoPhraseSetCurr,
                PseudoPhraseSetRes,
                [non_invariant_parts_for_given_invariant(InvariantPartRes,
                                                         TaxonReprRes)|
                  InvariantParts]): –
    taxon_formation_for_given_invariant(Incoincident_part,
                                        InvariantPartRes,
                                        PseudoPhraseSetCurr,
                                        PseudoPhraseSetNext,
                                        [Incoincident_part],
                                        InvariantsListRes,
                                        [ ], _),
    list_len(InvariantsListRes,InvariantsListRes_Len),
    InvariantsListRes_Len>1,!,
    taxon_transforming_respecting_new_invariant(InvariantPartRes,
                                                InvariantsListRes,
                                                TaxonReprRes),
    taxon_formation_for_given_pseudophrase(PseudoPhrase,
                                           PseudoPhraseSetNext,
                                           PseudoPhraseSetRes,
                                           InvariantParts).

taxon_formation_for_given_pseudophrase([_|PseudoPhrase],
                                       PseudoPhraseSetCurr,
                                       PseudoPhraseSetRes,
                                       InvariantParts): –
```

```
    taxon_formation_for_given_pseudophrase(PseudoPhrase,
                                           PseudoPhraseSetCurr,
                                           PseudoPhraseSetRes,
                                           InvariantParts).

/* Формирование множества таксонов для заданного множества
   буквенных инвариантов */

  taxons_formation_for_given_pseudophrases_set([ ], [ ]).

  taxons_formation_for_given_pseudophrases_set(
                                [PseudoPhrase|PseudoPhraseSetCurr],
                                InvariantParts): –
    not(PseudoPhrase=[]),!,
    taxon_formation_for_given_pseudophrase(PseudoPhrase,
                                           PseudoPhraseSetCurr,
                                           PseudoPhraseSetNext,
                                           InvariantParts1),
    taxons_formation_for_given_pseudophrases_set(
                                           PseudoPhraseSetNext,
                                           InvariantParts2),
    append(InvariantParts1,InvariantParts2,InvariantParts).

  taxons_formation_for_given_pseudophrases_set(
                                          [[ ]|PseudoPhraseSetCurr],
                                          InvariantParts): –
    taxons_formation_for_given_pseudophrases_set(
                                           PseudoPhraseSetCurr,
                                           InvariantParts).
```

```
/* Нумерация выявленных буквенных инвариантов для дальнейшего
   использования */

   invariants_numbering_for_given_non_invariant_parts(_, [ ], [ ]).

   invariants_numbering_for_given_non_invariant_parts(
                              Curr_Counter_of_coincidents,
                              [non_invariant_parts_for_given_invariant(
                                                   InvariantPartRes,_)|
                               InvariantDescrs],
                              [invariant_part(Curr_Counter_of_coincidents,
                                        InvariantPartRes)|InvariantParts]): –
      Next_Counter_of_coincidents=Curr_Counter_of_coincidents+1,
      invariants_numbering_for_given_non_invariant_parts(
                                        Next_Counter_of_coincidents,
                                        InvariantDescrs, InvariantParts).

/* Построение множества номеров буквенных инвариантов для
   последующего исследования частотных характеристик */

   invariants_numbers_gather([ ], [ ]).

   invariants_numbers_gather([invariant_part(Label,_Invariant)|
                                        InvariantParts], Res): –
      invariants_numbers_gather(InvariantParts,Res1),
      put(Label,Res1,Res).

/* Преобразования исходного множества фраз в соответствии с
   выявленными таксонами – НАЧАЛО */
```

```
/* Название “wdtnipfic” есть сокращение от “word description transform
   non invariant parts for invariants considering” (англ.). */

   wdtnipfic(word_considering(0,[ ],Incoincident_part,"false"),
                [non_invariant_parts_for_given_invariant(
                                                SymbolsForInvariantPart,
                                                PotentialEndings)|
                _InvariantPartsWithEndings],
                NumberedInvariantParts,
                word_considering(Label, SymbolsForInvariantPart,
                                        SomeEnding, "true")): –
       member(SomeEnding,PotentialEndings),
       append(SymbolsForInvariantPart,SomeEnding,Incoincident_part),
       member(invariant_part(Label,SymbolsForInvariantPart),
                NumberedInvariantParts).

   wdtnipfic(WordReprInit, [_|InvariantPartsWithEndings],
                NumberedInvariantParts, WordReprRes): –
       wdtnipfic(WordReprInit, InvariantPartsWithEndings,
                NumberedInvariantParts, WordReprRes).

/* Название “ptnipfic” есть сокращение от “phrase transform non
    invariant parts for invariants considering” (англ.). */

   ptnipfic([ ],_, _,[ ]).

   ptnipfic([WordReprInit|PhraseReprInit], InvariantPartsWithEndings,
            NumberedInvariantParts,
            [WordReprTr|PhraseReprTr]): –
```

```
    wdtnipfic(WordReprInit, InvariantPartsWithEndings,
                NumberedInvariantParts, WordReprTr),
    ptnipfic(PhraseReprInit,InvariantPartsWithEndings,
                NumberedInvariantParts,PhraseReprTr).

ptnipfic([WordReprInit|PhraseReprInit], InvariantPartsWithEndings,
            NumberedInvariantParts, [WordReprInit|PhraseReprTr]): –
    not(wdtnipfic(WordReprInit, InvariantPartsWithEndings,
                NumberedInvariantParts, _)),
    ptnipfic(PhraseReprInit, InvariantPartsWithEndings,
                NumberedInvariantParts, PhraseReprTr).

/* Название “pstnipfic” есть сокращение от “phrases transform non
   invariant parts for invariants considering” (англ.). */

pstnipfic([ ],_,_,[ ]).

pstnipfic([PhraseInit|PhrasesInit], InvariantPartsWithEndings,
            NumberedInvariantParts, [PhraseTr|PhrasesTr]): –
    ptnipfic(PhraseInit,
                InvariantPartsWithEndings,
                NumberedInvariantParts, PhraseTr),
    pstnipfic(PhrasesInit,
                InvariantPartsWithEndings,
                NumberedInvariantParts, PhrasesTr).

/* Преобразования исходного множества фраз в соответствии с
   выявленными таксонами – КОНЕЦ */
```

```
/* Вычисление частоты встречаемости элемента в списке */

   frequency_of_occurence(_, [ ], 0).

   frequency_of_occurence(Elem,[Lst|T],Freq): –
        member(Elem,Lst),
        frequency_of_occurence(Elem,T,Freq1),
        Freq=Freq1+1.

   frequency_of_occurence(Elem,[Lst|T],Freq): –
        not(member(Elem,Lst)),
        frequency_of_occurence(Elem,T,Freq).

/* Вычисление частоты встречаемости каждого элемента из списка */

   frequencies_of_occurence([ ], _, [ ]).

   frequencies_of_occurence([Elem|Others], Lst,
            [word_considering_aux_incoincident(Elem,Freq)|FrqsOcr]): –
        frequency_of_occurence(Elem,Lst,Freq),
        frequencies_of_occurence(Others,Lst,FrqsOcr).

/* Отобрать списки, включающие заданные элементы */

   orders_set_for_most_significant_index(_, [ ], [ ]).

   orders_set_for_most_significant_index(GivenIndexes,
                                [IndexList|ListOfIndexLists],
                                [IndexList|IndexListsForGivenIndexPresense]): –
```

```
    sub_set(GivenIndexes,IndexList),
    orders_set_for_most_significant_index(GivenIndexes,
                                          ListOfIndexLists,
                                          IndexListsForGivenIndexPresense).

orders_set_for_most_significant_index(GivenIndexes,
                                      [IndexList|ListOfIndexLists],
                                      IndexListsForGivenIndexPresense): –
    not(sub_set(GivenIndexes,IndexList)),
    orders_set_for_most_significant_index(GivenIndexes,
                                          ListOfIndexLists,
                                          IndexListsForGivenIndexPresense).

orders_set_for_most_significant_indexes(IndexesForSearch,
                                        _, [ ], _,
                                        IndexesForSearch).

orders_set_for_most_significant_indexes(GivenIndexesPrev, Estimation,
        [word_considering_aux_incoincident(MostSignificantIndex,
                                           _)|_],
        ListOfIndexListsPrev, IndexesForSearch): –
    orders_set_for_most_significant_index(
                                [MostSignificantIndex|GivenIndexesPrev],
                                ListOfIndexListsPrev,
                                ListOfIndexListsNext),
    list_len([MostSignificantIndex|GivenIndexesPrev],
             Significant_indexes_number),
    list_len(ListOfIndexListsNext,Number_of_Phrases),
    EstimationNext=Significant_indexes_number*Number_of_Phrases,
```

```
        EstimationNext<Estimation,
        IndexesForSearch=GivenIndexesPrev.

    orders_set_for_most_significant_indexes(GivenIndexesPrev, Estimation,
                [word_considering_aux_incoincident(MostSignificantIndex,
                                                    _)|FrqsOcr],
                ListOfIndexListsPrev, IndexesForSearch): –
        orders_set_for_most_significant_index(
                                [MostSignificantIndex|GivenIndexesPrev],
                                ListOfIndexListsPrev,
                                ListOfIndexListsNext),
        list_len([MostSignificantIndex|GivenIndexesPrev],
                Significant_indexes_number),
        list_len(ListOfIndexListsNext,Number_of_Phrases),
        EstimationNext=Significant_indexes_number*Number_of_Phrases,
        EstimationNext>=Estimation,
        orders_set_for_most_significant_indexes(
                                [MostSignificantIndex|GivenIndexesPrev],
                                EstimationNext, FrqsOcr,
                                ListOfIndexListsNext, IndexesForSearch).

/* Выделение подмножества индексов с наибольшей совокупной
    частотой встречаемости */

    most_significant_indexes_reveal(Indexes,
                                    OrdersSet,
                                    IndexesForSearch): –
        frequencies_of_occurence(Indexes, OrdersSet, FrqsOcrUnsorted),
        sort_hoar10(FrqsOcrUnsorted,FrqsOcr),
```

```
        orders_set_for_most_significant_indexes([ ], 0,
                                                FrqsOcr,
                                                OrdersSet,
                                                IndexesForSearch).

/* Разделение ситуации СЭ - КОНЕЦ */

/* Сравнение пары фраз – НАЧАЛО */

    word_and_phrase_processing(Counter,_,[ ],[ ],[ ],Counter).

    word_and_phrase_processing(Counter, Incoincident_part1,
                    [word_considering(0,[ ],Incoincident_part2,"false")|
                      SentenceRepr],
                    [word_considering_aux(Counter,
                                          Conterminous_part,
                                          New_Incoincident_part1,
                                          Incoincident_part1)|Aux1],
                    [word_considering_aux(Counter,
                                          Conterminous_part,
                                          New_Incoincident_part2,
                                          Incoincident_part2)|Aux2],
                    CounterRes): –
        common_prefix(Incoincident_part1,
                      Incoincident_part2,
                      Conterminous_part),
        prefix(Conterminous_part,
               Incoincident_part1,
               New_Incoincident_part1),
```

```
    prefix(Conterminous_part,
           Incoincident_part2,
           New_Incoincident_part2),
    not(Conterminous_part=[ ]),
    Counter1=Counter+1,
    word_and_phrase_processing(Counter1,
                               Incoincident_part1,
                               SentenceRepr,
                               Aux1,
                               Aux2,
                               CounterRes).

word_and_phrase_processing(Counter, Incoincident_part1,
                           [_|SentenceRepr],
                           Aux1, Aux2,
                           CounterRes): –
    word_and_phrase_processing(Counter,
                               Incoincident_part1,
                               SentenceRepr,
                               Aux1,
                               Aux2,
                               CounterRes).

pair_of_phrases_processing2([word_considering(0,[ ],
                                              Incoincident_part1, "false")|
                             Sentence1Repr],
                            Sentence2_curr_considering,
                            Aux1,
                            Aux2): –
```

```
    word_and_phrase_processing(0, Incoincident_part1,
                                  Sentence2_curr_considering,
                                  Aux11, Aux22, CounterNext),
    pair_of_phrases_processing1(CounterNext,Aux22,
                                [word_considering(0, [ ],
                                                  Incoincident_part1,
                                                  "false")|
                                 Sentence1Repr],
                                Aux222, Aux111),
    unit_sets(Aux11,Aux111,Aux1),
    unit_sets(Aux22,Aux222,Aux2).

pair_of_phrases_processing1(_,[ ],_,[ ],[ ]).

pair_of_phrases_processing1(Counter,
                            [word_considering_aux(_,
                                                  _Conterminous_part,
                                                  _New_Incoincident_part2,
                                                  Incoincident_part2)|Aux2],
                            SentenceRepr,
                            Aux11,
                            Aux22): –
    word_and_phrase_processing(Counter, Incoincident_part2,
                                  SentenceRepr,
                                  Aux111, Aux222,
                                  CounterNext),
    pair_of_phrases_processing1(CounterNext, Aux2,
                                SentenceRepr,
                                Aux1111, Aux2222),
```

```
    unit_sets(Aux111,
                  Aux1111,
                  Aux11),
    unit_sets(Aux222,
                  Aux2222,
                  Aux22).

gather_words_from_word_considering_aux([ ], [ ]).

gather_words_from_word_considering_aux(
                  [word_considering_aux(_,_,_,Word_char_list)|AuList],
                  [Word_char_list|Word_char_lists]): –
    gather_words_from_word_considering_aux(AuList,
                                                Word_char_lists).

word_considering_aux_incoincident_estimate([ ], [ ], [ ]).

word_considering_aux_incoincident_estimate(
        [word_considering_aux(Counter, Conterminous_part,
                                    New_Incoincident_part1, _)|Aux1],
        [word_considering_aux(Counter, Conterminous_part,
                                    New_Incoincident_part2, _)|Aux2],
        [word_considering_aux_incoincident(Counter,Diff)|Aux3]): –
    list_len(New_Incoincident_part1,New_Incoincident_part1_len),
    list_len(New_Incoincident_part2,New_Incoincident_part2_len),
    Diff=New_Incoincident_part1_len+New_Incoincident_part2_len,
    word_considering_aux_incoincident_estimate(Aux1,Aux2,Aux3).

select_by_estimations([ ], _, _, [ ], [ ]).
```

```
select_by_estimations(
                [word_considering_aux_incoincident(Counter,_)|Tail],
                Aux1, Aux2,
                [word_considering_aux(Counter, Conterminous_part,
                                      New_Incoincident_part1,
                                      Wrd1Chars)|NewAux1],
                [word_considering_aux(Counter, Conterminous_part,
                                      New_Incoincident_part2,
                                      Wrd2Chars)|NewAux2]): –
    member(word_considering_aux(Counter, Conterminous_part,
                                New_Incoincident_part1,
                                Wrd1Chars), Aux1),
    member(word_considering_aux(Counter,Conterminous_part,
                                New_Incoincident_part2,
                                Wrd2Chars), Aux2),
    select_by_estimations(Tail,Aux1,Aux2,NewAux1,NewAux2).

renumbering(MaxNumber,[ ] ,[ ] ,[ ], [ ] ,MaxNumber).

renumbering(NewNumber,
                [word_considering_aux(Counter,
                                      Conterminous_part,
                                      New_Incoincident_part1,
                                      Wrd1Chars)|Aux1],
                [word_considering_aux(Counter,
                                      Conterminous_part,
                                      New_Incoincident_part2,
                                      Wrd2Chars)|Aux2],
```

```
                [word_considering_aux(NewNumber,
                                      Conterminous_part,
                                      New_Incoincident_part1,
                                      Wrd1Chars)|NewAux1],
                [word_considering_aux(NewNumber,
                                      Conterminous_part,
                                      New_Incoincident_part2,
                                      Wrd2Chars)|NewAux2],
                MaxNumber): –
    NewNumber1=NewNumber+1,
    renumbering(NewNumber1,
                Aux1, Aux2, NewAux1,
                NewAux2, MaxNumber).

setting_revealed_conformities(_, [ ], [ ],
                              Sentence1ReprRes,
                              Sentence2ReprRes,
                              Sentence1ReprRes,
                              Sentence2ReprRes).

setting_revealed_conformities(Flag,
                              [Aux1Head|Aux1Tail],
                              [Aux2Head|Aux2Tail],
                              Sentence1ReprOld, Sentence2ReprOld,
                              Sentence1ReprRes, Sentence2ReprRes): –
    setting_revealed_conformity(Flag,
                                Aux1Head,
                                Sentence1ReprOld,
                                Sentence1ReprNew),
```

```
    setting_revealed_conformity(Flag,Aux2Head,Sentence2ReprOld,
                                Sentence2ReprNew),
    setting_revealed_conformities(Flag,
                                  Aux1Tail, Aux2Tail,
                                  Sentence1ReprNew,
                                  Sentence2ReprNew,
                                  Sentence1ReprRes,
                                  Sentence2ReprRes).

setting_revealed_conformity(Flag,
                            word_considering_aux(NewLabel,
                                                 Conterminous_part,
                                                 New_Incoincident_part,
                                                 WrdChars),
                            [word_considering(0,[],WrdChars,"false")|
                              SentenceRepr],
                            [word_considering(NewLabel,
                                                 Conterminous_part,
                                                 New_Incoincident_part,
                                                 Flag)|
                              SentenceRepr]): –!.

setting_revealed_conformity(Flag,
                            word_considering_aux(NewLabel,
                                                 Conterminous_part,
                                                 New_Incoincident_part,
                                                 WrdChars),
                            [H|SentenceRepr],
                            [H|SentenceReprNew]): –
```

```
    setting_revealed_conformity(Flag,
                                word_considering_aux(NewLabel,
                                                     Conterminous_part,
                                                     New_Incoincident_part,
                                                     WrdChars),
                                SentenceRepr,
                                SentenceReprNew).

pair_of_phrases_processing(_,
                           New_Counter_of_coincidents,
                           0, Ph1, Ph2, Ph1, Ph2,
                           New_Counter_of_coincidents).


pair_of_phrases_processing(Flag,
                           Counter_of_coincidents,
                           Words_must_be_considered,
                           Ph1_for_test,
                           Ph2_for_test,
                           Ph1_res,
                           Ph2_res,
                           Res_Counter_of_coincidents): –
    Words_must_be_considered>0,
    pair_of_phrases_processing2(Ph1_for_test,
                                Ph2_for_test,
                                Aux1_unsorted,
                                Aux2_unsorted),
    sort_hoar1(Aux1_unsorted,Aux1),
    sort_hoar1(Aux2_unsorted,Aux2),
```

```
word_considering_aux_incoincident_estimate(Aux1,
                                           Aux2,
                                           Aux3_unsorted),
sort_hoar2(Aux3_unsorted,Aux3),
gather_words_from_word_considering_aux(Aux1,WrdsAux1),
list_set(WrdsAux1,WrdsSetAux1),
gather_words_from_word_considering_aux(Aux2,WrdsAux2),
list_set(WrdsAux2,WrdsSetAux2),
list_len(WrdsSetAux1,WrdsSetLenAux1),
list_len(WrdsSetAux2,WrdsSetLenAux2),
min(WrdsSetLenAux1,WrdsSetLenAux2,WrdsSetMinLen),
first_n(Aux3,WrdsSetMinLen,Mins_from_Aux3,_),
select_by_estimations(Mins_from_Aux3,
                      Aux1,
                      Aux2,
                      NewAux1,
                      NewAux2),
sort_hoar1(NewAux1,X),
sort_hoar1(NewAux2,Y),
renumbering(Counter_of_coincidents,
            X,Y,
            X1,Y1,
            New_Counter_of_coincidents),
setting_revealed_conformities(Flag,X1,Y1,
                              Ph1_for_test,Ph2_for_test,
                              Ph1_new,Ph2_new),
Ph1_new=[Head_Ph1_new1|Tail_Ph1_new1],
append(Tail_Ph1_new1,[Head_Ph1_new1],Ph1_for_test_new),
Words_must_be_considered1=Words_must_be_considered−1,
```

```
        pair_of_phrases_processing(Flag,
                                   New_Counter_of_coincidents,
                                   Words_must_be_considered1,
                                   Ph1_for_test_new, Ph2_new,
                                   Ph1_res, Ph2_res,
                                   Res_Counter_of_coincidents).

    pair_of_phrases_processing(Flag,
                               Counter_of_coincidents,
                               Words_must_be_considered,
                               Ph1_for_test, Ph2_for_test,
                               Ph1_res, Ph2_res,
                               Res_Counter_of_coincidents): –
        Words_must_be_considered>0,
        not(pair_of_phrases_processing2(Ph1_for_test,Ph2_for_test,_,_)),
        Ph1_for_test=[Head_Ph1_for_test|Tail_Ph1_for_test],
        append(Tail_Ph1_for_test,[Head_Ph1_for_test],Ph1_for_test_new),
        Words_must_be_considered1=Words_must_be_considered – 1,
        pair_of_phrases_processing(Flag,
                                   Counter_of_coincidents,
                                   Words_must_be_considered1,
                                   Ph1_for_test_new,
                                   Ph2_for_test,
                                   Ph1_res,
                                   Ph2_res,
                                   Res_Counter_of_coincidents).

/* Сравнение пары фраз - КОНЕЦ */
```

```
/* Сбор информации о порядках слов относительно выявленного
   инварианта по предложениям */

orders_of_words_in_sentences([ ], [ ]).

orders_of_words_in_sentences([Sentence|Sentences],
                             [Order|Orders]): –
    order_of_words_in_sentence(Sentence,Order),
    orders_of_words_in_sentences(Sentences,Orders).

order_of_words_in_sentence([ ], [ ]).

order_of_words_in_sentence([word_considering(Label,
                                             _Conterminous_part,
                                             _Incoincident_part,
                                             Flag)|RestOfPhrase],
                           [Label|LabelList]): –
    Flag="true",
    order_of_words_in_sentence(RestOfPhrase,LabelList).

order_of_words_in_sentence([word_considering(_Label,
                                             _Conterminous_part,
                                             _Incoincident_part,
                                             Flag)|RestOfPhrase],
                           LabelList): –
    not(Flag="true"),
    order_of_words_in_sentence(RestOfPhrase,LabelList).
```

```
/* Построение списка частей слова, не меняющихся при взаимном
   синонимическом преобразовании фраз внутри пары.
   Далее при обработке перифраз список инвариантов не пополняется,
   с целью выявления устойчивых сочетаний слов из него могут
   удаляться инварианты, не нашедшие прообразов в новых
   перифразах */

   invariant_part_list_building_for_pair(_,[ ],[ ]).

   invariant_part_list_building_for_pair(Flag,
                                         [word_considering(0,[],_,"false")|SentenceRepr],
                                         Invariant_parts_list): –
       invariant_part_list_building_for_pair(Flag, SentenceRepr,
                                             Invariant_parts_list).

   invariant_part_list_building_for_pair(Flag,
                                         [word_considering(NewLabel,
                                                           Conterminous_part,
                                                           _,
                                                           Flag)|SentenceRepr],
                                         [invariant_part(NewLabel,
                                                         Conterminous_part)|
                                                         Invariant_parts_list]): –
       invariant_part_list_building_for_pair(Flag, SentenceRepr,
                                             Invariant_parts_list).

/* Проверка множества фраз с уточнением инварианта */

   phrases_check_by_invariant(_,[ ],Invariant,Invariant,[ ]).
```

```
phrases_check_by_invariant(Flag,[Phrase|PhrasesSet],
                           CurrInvar,ResInvar,
                           [PhraseReprRes|PhrasesSetReprRes]): –
    phrase_check_by_invariant(Flag, CurrInvar, Phrase,
                              PhraseReprRes,InvarForNext),
    phrases_check_by_invariant(Flag,PhrasesSet,InvarForNext,
                              ResInvar,PhrasesSetReprRes).

/* Проверка очередной фразы по выявленному инварианту */
/* Название “pcbiaptnit” есть сокращение от “phrase check by invariant
   and pseudophrase to new invariant transform” (англ.). */

pcbiaptnit("local", Sentence2_curr_considering, Invariant_parts_list,
           Sentence2_res_considering, Invariant_parts_list_new,
           Invariant_parts_list_res): –
    Invariant_parts_list_new=[ ],!,
    Sentence2_res_considering=Sentence2_curr_considering,
    Invariant_parts_list_res=Invariant_parts_list.

pcbiaptnit(Flag, Sentence2_curr_considering, _Invariant_parts_list,
           Sentence2_res_considering, Invariant_parts_list_new,
           Invariant_parts_list_res): –
    phrase_check_by_new_invariant(Flag,
                                  Invariant_parts_list_new,
                                  Sentence2_curr_considering,
                                  Sentence2_res_considering),
    Invariant_parts_list_res=Invariant_parts_list_new.
```

```
phrase_check_by_invariant(Flag, Invariant_parts_list,
                          Sentence2_curr_considering,
                          Sentence2_res_considering,
                          Invariant_parts_list_res): –
    invariant_to_pseudophrase_transform(Invariant_parts_list,
                                        Pseudo_Phrase),
    list_len(Pseudo_Phrase,Pseudo_Phrase_LEN),
    pair_of_phrases_processing(Flag,1, Pseudo_Phrase_LEN,
                Pseudo_Phrase,Sentence2_curr_considering,
                Pseudo_Phrase_new,Sentence2_next_considering,_),
    pseudophrase_to_new_invariant_transform(Flag,
                                        Invariant_parts_list,
                                        Pseudo_Phrase_new,
                                        Invariant_parts_list_new),
    pcbiaptnit(Flag, Sentence2_next_considering, Invariant_parts_list,
            Sentence2_res_considering, Invariant_parts_list_new,
            Invariant_parts_list_res).

/* Генерация псевдофразы для списка инвариантов */

invariant_to_pseudophrase_transform([ ], [ ]).

invariant_to_pseudophrase_transform(
                [invariant_part(_Label,Conterminous_part)|
                                        Invariant_parts_list],
                [word_considering(0,[],Conterminous_part,"false")|
                                        SentenceRepr]): –
    invariant_to_pseudophrase_transform(Invariant_parts_list,
                                        SentenceRepr).
```

```
/* Новый инвариант и расстановка композиционных меток */

   search_pseudophrase_for_invariant(Flag,
                                     [word_considering(_,New_Conterminous_part,
                                                       New_Incoincident_part,Flag)|
                                      _SentenceReprRest],
                                     Conterminous_part, New_Conterminous_part): –
       append(New_Conterminous_part,
              New_Incoincident_part,
              Conterminous_part),!.

   search_pseudophrase_for_invariant(Flag, [_|SentenceReprRest],
                                           Conterminous_part,
                                           New_Conterminous_part): –
       search_pseudophrase_for_invariant(Flag,SentenceReprRest,
                                         Conterminous_part,
                                         New_Conterminous_part).

   pseudophrase_to_new_invariant_transform(_, [ ], _, [ ]).

   pseudophrase_to_new_invariant_transform(Flag,
                                           [invariant_part(Label,Conterminous_part)|
                                                       Invariant_parts_list_old],
                                           SentenceRepr,
                                           [invariant_part(Label,New_Conterminous_part)|
                                                       Invariant_parts_list_new]): –
       search_pseudophrase_for_invariant(Flag,SentenceRepr,
                                         Conterminous_part,
                                         New_Conterminous_part),
```

```
    pseudophrase_to_new_invariant_transform(Flag,
                                Invariant_parts_list_old, SentenceRepr,
                                Invariant_parts_list_new).

pseudophrase_to_new_invariant_transform(Flag,
                                [invariant_part(_,Conterminous_part)|
                                        Invariant_parts_list_old],
                                SentenceRepr,
                                Invariant_parts_list_new): –
    not(search_pseudophrase_for_invariant(Flag, SentenceRepr,
                                        Conterminous_part,_)),
    pseudophrase_to_new_invariant_transform(Flag,
                                        Invariant_parts_list_old,
                                        SentenceRepr,
                                        Invariant_parts_list_new).

phrase_check_by_new_invariant_word(Flag,
                            invariant_part(Label,Conterminous_part),
                            [word_considering(_,Conterminous_part,
                                        Incoincident_part,Flag)|
                              SentenceReprRest],
                            [word_considering(Label,Conterminous_part,
                                        Incoincident_part,Flag)|
                              SentenceReprRest]): – !.

phrase_check_by_new_invariant_word(Flag,
                            invariant_part(Label,Conterminous_part),
                            [Wrd_Consider|SentenceReprRest],
                            [Wrd_Consider|SentenceReprRestNew]): –
```

```
    phrase_check_by_new_invariant_word(Flag,
                                       invariant_part(Label,Conterminous_part),
                                       SentenceReprRest,
                                       SentenceReprRestNew).

phrase_check_by_new_invariant(_,[ ], SentReprRes,SentReprRes).

phrase_check_by_new_invariant(Flag, [Invariant|Invariants],
                                    SentReprCurr, SentReprRes): –
    phrase_check_by_new_invariant_word(Flag, Invariant,
                                             SentReprCurr,
                                             SentReprNext),
    phrase_check_by_new_invariant(Flag, Invariants,
                                        SentReprNext,
                                        SentReprRes).

/* Окончательное преобразование СЭ-множества с учетом
   выявленного инварианта */

word_transform_invariant_respecting(Flag,
                    word_considering(_,Conterminous_part,
                                       Incoincident_part,_),
                    [ ],
                    word_considering(0, [ ],
                                       Incoincident_part_new,
                                       "false")): –
    not(Flag="false"),
    append(Conterminous_part, Incoincident_part,
           Incoincident_part_new),!.
```

```
word_transform_invariant_respecting(Flag,
                                    word_considering(Label,
                                                     Conterminous_part,
                                                     Incoincident_part,
                                                     Flag),
                                    [invariant_part(Label,
                                                    Conterminous_part)|
                                     _Invariant_parts],
                                    word_considering(Label,
                                                     Conterminous_part,
                                                     Incoincident_part,
                                                     Flag)): – !.

word_transform_invariant_respecting(Flag,
                                      word_considering(Label,
                                                       Conterminous_part,
                                                       Incoincident_part,
                                                       Flag),
                                      [invariant_part(Label,
                                                      Conterminous_part_new)|
                                       _Invariant_parts],
                                      word_considering(Label,
                                                       Conterminous_part_new,
                                                       Incoincident_part_new,
                                                       Flag)): –
    append(Conterminous_part_new, Incoincident_part_for_add,
           Conterminous_part),!,
    append(Incoincident_part_for_add, Incoincident_part,
           Incoincident_part_new).
```

```
word_transform_invariant_respecting(Flag,
                                    word_considering(Label,
                                                     Conterminous_part,
                                                     Incoincident_part,
                                                     Flag),
                                    [invariant_part(Label1,_)|
                                     Invariant_parts],
                                    word_considering(Label_Res,
                                                     Conterminous_part_res,
                                                     Incoincident_part_res,
                                                     Flag_res)): –
    not(Label=Label1),
    word_transform_invariant_respecting(Flag,
                                        word_considering(Label,
                                                         Conterminous_part,
                                                         Incoincident_part,
                                                         Flag),
                                        Invariant_parts,
                                        word_considering(Label_Res,
                                                         Conterminous_part_res,
                                                         Incoincident_part_res,
                                        Flag_res)).

word_transform_invariant_respecting(Flag,
                                    word_considering(Label,
                                                     Conterminous_part,
                                                     Incoincident_part,
                                                     "false"),
                                    _,
```

```
                                word_considering(Label,
                                                 Conterminous_part,
                                                 Incoincident_part,
                                                 "false")): –
    not(Flag="false").

phrase_transform_invariant_respecting(_, [ ], _, [ ]).

phrase_transform_invariant_respecting(Flag,
                                      [WordRepr|PhraseRest],
                                      Invar,
                                      [WordReprNew|PhraseRestNew]): –
    word_transform_invariant_respecting(Flag,
                                        WordRepr,Invar,
                                        WordReprNew),
    phrase_transform_invariant_respecting(Flag,
                                          PhraseRest,Invar,
                                          PhraseRestNew).

phrases_transform_invariant_respecting(_, [ ], _, [ ]).

phrases_transform_invariant_respecting(Flag,
                                       [Phrase|PhrasesSet],
                                       Invar,
                                       [NewPhrase|NewPhrasesSet]): –
    phrase_transform_invariant_respecting(Flag,
                                          Phrase,
                                          Invar,
                                          NewPhrase),
```

```
        phrases_transform_invariant_respecting(Flag, PhrasesSet,
                                                Invar, NewPhrasesSet).

/* Реализация вспомогательных процедур */
/* Является ли один список префиксом другого */

   prefix([ ], Suffix, Suffix).

   prefix([H|T],[H|T1],Suffix): – prefix(T,T1,Suffix).

/* Имеют ли два списка общий префикс */

   common_prefix([ ],_,[ ]).

   common_prefix(_,[ ],[ ]).

   common_prefix([H1|_],[H2|_],[ ]): – not(H1=H2).

   common_prefix([H|T1],[H|T2],[H|Res]): – common_prefix(T1,T2,Res).

/* Нахождение длины списка */

   list_len([ ],0).
   list_len([_|Tail],Len): – list_len(Tail,Len1), Len=Len1+1.

/* Объединение двух списков */

   append([ ],L,L).
   append([Head|Tail],Lst2,[Head|Tail_res]): – append(Tail,Lst2,Tail_res).
```

```
/* Принадлежность элемента списку */

   member(Head,[Head|_]).
   member(Elem,[_|T]): – member(Elem,T).

/* Удаление всех вхождений заданного элемента в список */

   delete(_,[ ],[ ]).
   delete(H,[H|T],Res): – delete(H,T,Res).
   delete(Elem,[H|T],[H|Res]): – not(H=Elem), delete(Elem,T,Res).

/* Преобразование списка в множество */

   list_set([ ],[ ]).

   list_set([Head_lst|Tail_lst],Res): –
        member(Head_lst,Tail_lst),!, list_set(Tail_lst,Res).

   list_set([Head_lst|Tail_lst],[Head_lst|Res]): –
        list_set(Tail_lst,Res).

/* Выделение заданного количества первых элементов списка */

   first_n(Lst,N,Lst,[ ]): – list_len(Lst,ListLen), N>=ListLen,!.

   first_n(Lst,0,[ ],Lst).

   first_n([Lst_Head|Lst_Tail],N,[Lst_Head|Rest_of_First],Rest_of_List): –
        N>0, N1=N–1, first_n(Lst_Tail,N1,Rest_of_First,Rest_of_List).
```

```
/* Объединение множеств */

   unit_sets([ ],Set2,Set2).
   unit_sets([H|T],Set2,[H|Res]): –
        not(member(H,Set2)),
        unit_sets(T,Set2,Res).

   unit_sets([H|T],Set2,Res): –
        member(H,Set2),
        unit_sets(T,Set2,Res).

/* Правило помещает объект в список, если этот объект там
     отсутствует */

   put(Obj,Arg,[Obj|Arg]): – not(member(Obj,Arg)).

   put(Obj,Arg,Arg): – member(Obj,Arg).

/* Является ли одно множество подмножеством другого */

   sub_set([ ],_).
   sub_set([H|T],Set2):  – member(H,Set2), sub_set(T,Set2).

/* Минимум из двух чисел */

   min(X,Y,X): – X<Y.
   min(X,Y,Y): – X>=Y.
```

```
/* Сортировка Хаара */

   sort_hoar1([ ],[ ]).

   sort_hoar1([word_considering_aux(Counter,
                                    Common,Incoinc,
                                    WordTotal)|Tail],
              Res): –
      partition1(Tail,
                 word_considering_aux(Counter,
                                      Common,Incoinc,WordTotal),
                 Littles,
                 Bigs),
      sort_hoar1(Littles,Ls),
      sort_hoar1(Bigs,Bs),
      append(Ls,
             [word_considering_aux(Counter,
                                   Common,Incoinc,WordTotal)|Bs],
             Res).

   sort_hoar2([ ],[ ]).

   sort_hoar2([word_considering_aux_incoincident(Counter,
                                                 IncoincNum)|Tail],Res): –
      partition2(Tail,
                 word_considering_aux_incoincident(Counter,
                                                   IncoincNum),
                 Littles,
                 Bigs),
```

```
    sort_hoar2(Littles,Ls),
    sort_hoar2(Bigs,Bs),
    append(Ls,
           [word_considering_aux_incoincident(Counter,
                                              IncoincNum)|Bs],
           Res).

sort_hoar10([ ],[ ]).

sort_hoar10([word_considering_aux_incoincident(Counter,
                                               IncoincNum)|Tail],
            Res): –
    partition2(Tail,
               word_considering_aux_incoincident(Counter,
                                                 IncoincNum),
               Littles,
               Bigs),
    sort_hoar10(Littles,Ls),
    sort_hoar10(Bigs,Bs),
    append(Bs,
           [word_considering_aux_incoincident(Counter,
                                              IncoincNum)|Ls],
           Res).

sort_hoar11([ ],[ ]).

sort_hoar11([Head|Tail],Res): –
    partition9(Tail,Head,Littles,Bigs),
    sort_hoar11(Littles,Ls),
```

```
        sort_hoar11(Bigs,Bs),
        append(Ls,[Head|Bs],Res).

/* Разделение списка на "большие" и "меньшие" относительно
   заданного барьера */

   partition1([ ],_,[ ],[ ]).

   partition1([word_considering_aux(Counter,
                                    Common,Incoinc,
                                    WordTotal)|Tail],
              word_considering_aux(Barier,BCommon,
                                    BIncoinc,BWordTotal),
              [word_considering_aux(Counter,Common,
                                    Incoinc,WordTotal)|Littles],
              Bigs): –
        Counter<=Barier,
        partition1(Tail,
                   word_considering_aux(Barier,BCommon,
                                        BIncoinc,BWordTotal),
                   Littles, Bigs).

   partition1([word_considering_aux(Counter,
                                    Common,Incoinc,
                                    WordTotal)|Tail],
              word_considering_aux(Barier,
                                    BCommon,
                                    BIncoinc,
                                    BWordTotal),
```

```
            Littles,
            [word_considering_aux(Counter,
                                  Common,
                                  Incoinc,
                                  WordTotal)|Bigs]): –
    Counter>Barier,
    partition1(Tail,
               word_considering_aux(Barier,
                                    BCommon,
                                    BIncoinc,
                                    BWordTotal),
               Littles,
               Bigs).

partition2([ ],_,[ ],[ ]).

partition2([word_considering_aux_incoincident(Counter,
                                              Incoinc)|Tail],
           word_considering_aux_incoincident(BCounter,BIncoinc),
           [word_considering_aux_incoincident(Counter,
                                              Incoinc)|Littles],
           Bigs): –
    Incoinc<=BIncoinc,
    partition2(Tail,
               word_considering_aux_incoincident(BCounter,
                                                 BIncoinc),
               Littles,
               Bigs).
```

```
    partition2([word_considering_aux_incoincident(Counter,Incoinc)|Tail],
                word_considering_aux_incoincident(Barier,BIncoinc),
                Littles,
                [word_considering_aux_incoincident(Counter,Incoinc)|Bigs]): –
        Incoinc>BIncoinc,
        partition2(Tail,
                    word_considering_aux_incoincident(Barier,BIncoinc),
                    Littles,
                    Bigs).

    partition9([ ],_,[ ],[ ]).

    partition9([Head|Tail], Barier, [Head|Littles], Bigs): –
        Head<=Barier, partition9(Tail,Barier,Littles,Bigs).

partition9([Head|Tail], Barier, Littles, [Head|Bigs]): –
        Head>Barier, partition9(Tail,Barier,Littles,Bigs).
```

СПИСОК СОКРАЩЕНИЙ

ЕЯ – Естественный язык

СЭ – Семантическая эквивалентность

РОСС – Русский общесемантический словарь

ЛФ – Лексическая функция

ГСС – Глубинная синтаксическая структура

СемП – Семантическое представление

СГ – Семантический граф

ТКС – Толково-комбинаторный словарь

АФП – Анализ формальных понятий

ФП – Формальное понятие

НОПП – Наибольшее общее подпонятие

НОСП – Наименьшее общее суперпонятие

ЛСК – Лексическая синонимическая конструкция

МУ – Модель управления

ЛЗ – Лексическое значение

ИГ – Именная группа

СК – Семантический класс

ХФ – Характеристическая функция

РЗ – Расщепленное значение

СХ – Семантическая характеристика

РПЗ – Расщепленное предикатное значение

ОГЛАВЛЕНИЕ